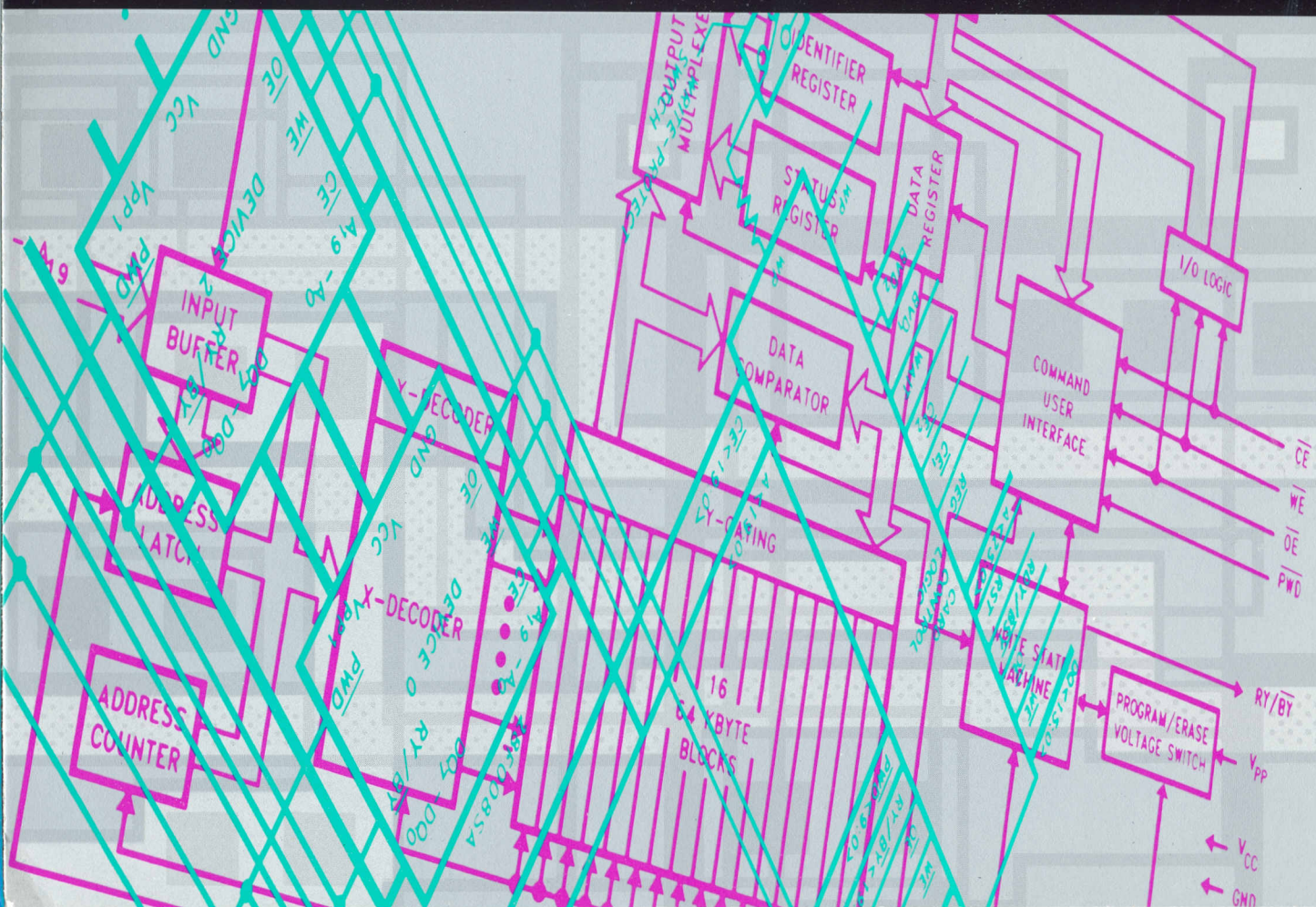


Flash Memory: Volume I

Boot Block
Components
Bulk-Erase
Components
FlashFile™
Components



intel®



LITERATURE

To order Intel literature or obtain literature pricing information in the U.S. and Canada call or write Intel Literature Sales. In Europe and other international locations, please contact your **local** sales office or distributor.

INTEL LITERATURE SALES
P.O. Box 7641
Mt. Prospect, IL 60056-7641

In the U.S. and Canada
call toll free
(800) 548-4725

This 800 number is for external customers only.

CURRENT HANDBOOKS

Product line handbooks contain data sheets, application notes, article reprints and other design information. All handbooks can be ordered individually, and most are available in a pre-packaged set in the U.S. and Canada.

Title	Intel Order Number	ISBN
SET OF FOURTEEN HANDBOOKS (Available in U.S. and Canada)	231003	N/A
CONTENTS LISTED BELOW FOR INDIVIDUAL ORDERING:		
CONNECTIVITY	231658	1-55512-202-7
EMBEDDED MICROCONTROLLERS	270646	1-55512-203-5
EMBEDDED MICROPROCESSORS	272396	1-55512-204-3
FLASH MEMORY (2 volume set)	210830	1-55512-214-0
MICROPROCESSORS, VOL. 1: Intel386™ 80286 & 8086 MICROPROCESSORS	230843	1-55512-196-9
MICROPROCESSORS, VOL. 2: Intel486™ MICROPROCESSORS	241731	1-55512-197-7
MICROPROCESSORS, VOL. 3: PENTIUM™ PROCESSORS	241732	1-55512-198-5
i750®, i860™, i960® PROCESSORS AND RELATED PRODUCTS	272084	1-55512-217-5
OEM BOARDS, SYSTEMS & SOFTWARE	280407	1-55512-201-9
PACKAGING	240800	1-55512-208-6
PERIPHERAL COMPONENTS	296467	1-55512-207-8
PRODUCT OVERVIEW	210846	N/A
PROGRAMMABLE LOGIC	296083	1-55512-206-X
NETWORKING	297360	1-55512-220-5
ADDITIONAL LITERATURE: (Not included in handbook set)		
AUTOMOTIVE PRODUCTS	231792	1-55512-212-4
COMPONENTS QUALITY/RELIABILITY	210997	1-55512-132-2
CUSTOMER LITERATURE GUIDE	210620	N/A
EMBEDDED APPLICATIONS (1993/94)	270648	1-55512-179-9
INTERNATIONAL LITERATURE GUIDE (Available in Europe only)	E00029	N/A
MILITARY AND SPECIAL PRODUCTS (2 volume set)	210461	1-55512-213-2
SYSTEMS QUALITY/RELIABILITY	231762	1-55512-046-6

Flash Memory: Volume I

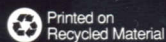
In today's vast array of semiconductor memory choices, the design engineer's job is to match memory characteristics to the application. Since 1971, Intel has offered a variety of memory devices to suit a wide range of applications. The most compelling of these is flash memory – a technology which Intel has remained the market-share leader for over six years.

Because of their inherent features, Intel's Flash Memory products are dramatically altering traditional computer memory solutions. Today's hand-held palmtop computers, personal organizers and notebook PCs need a solid-state storage technology that provides the optimum combination of performance, size, weight, low power and shock resistance demanded by today's mobile computer user. As a high density, non-volatile, read/write semiconductor technology, Intel Flash Memory is the ideal memory medium to meet these seemingly competing requirements whether implemented in memory cards, solid-state disks or components.

This handbook is devoted to techniques and information to help design semiconductor memory into an application or system. Informative data sheets provide many comprehensive charts, block diagrams, operating characteristics and programming modes. Application notes provide diagrams and hardware design information; relevant article reprints are also included.

intel®

Order Number: 210830-013
Printed in USA/194/30K/RRD/CC
Memory Products



ISBN 1-55512-200-0



9 781555 122003

90000



FLASH MEMORY VOLUME I

1994

Intel Corporation makes no warranty for the use of its products and assumes no responsibility for any errors which may appear in this document nor does it make a commitment to update the information contained herein.

Intel reserves the right to make changes to these specifications at any time, without notice.

Contact your local Intel sales office or your distributor to obtain the latest specifications before placing your product order.

Intel Corporation and Intel's FASTPATH are not affiliated with Kinetics, a division of Excerpt, Inc. or its FASTPATH trademark or products.

*Other brands and names are the property of their respective owners.

Additional copies of this document or other data literature may be obtained from:

Intel Corporation
Literature Sales
P.O. Box 7841
Mt. Pleasant, IL 60056-7841
or call 1-800-537-4853

©1994 INTEL CORPORATION



FLASH MEMORY VOLUME I

Intel Corporation makes no warranty for the use of its products and assumes no responsibility for any errors which may appear in this document nor does it make a commitment to update the information contained herein.

Intel retains the right to make changes to these specifications at any time, without notice.

Contact your local Intel sales office or your distributor to obtain the latest specifications before placing your product order.

MDS is an ordering code only and is not used as a product name or trademark of Intel Corporation.

Intel Corporation and Intel's FASTPATH are not affiliated with Kinetics, a division of Excelan, Inc. or its FASTPATH trademark or products.

*Other brands and names are the property of their respective owners.

Additional copies of this document or other Intel literature may be obtained from:

Intel Corporation
Literature Sales
P.O. Box 7641
Mt. Prospect, IL 60056-7641
or call 1-800-879-4683

©INTEL CORPORATION, 1993

LGCPY1/100693

DATA SHEET DESIGNATIONS

Intel uses various data sheet markings to designate each phase of the document as it relates to the product. The marking appears in the upper, right-hand corner of the data sheet. The following is the definition of these markings:

Data Sheet Marking	Description
Product Preview	Contains information on products in the design phase of development. Do not finalize a design with this information. Revised information will be published when the product becomes available.
Advanced Information	Contains information on products being sampled or in the initial production phase of development.*
Preliminary	Contains preliminary information on new products in production.*
No Marking	Contains information on products in full production.*

*Specifications within these data sheets are subject to change without notice. Verify with your local Intel sales office that you have the latest data sheet before finalizing a design.



Memory Overview

1

Flash Overview

2

FlashFile™ Components

3

Boot Block Components

4

Bulk-Erase Components

5

Memory Cards

6

Flash ATA Drives

7

Automotive Components

8

Process Engineering Reports

9

Process Engineering Reports

Automotive Components

Flash ATA Drives

Memory Cards

Bulk-Erase Components

Boot Block Components

FlashFile™ Components

Flash Overview

Memory Overview

Article Reprints

10

Table of Contents

Alphanumeric Index	xiv
CHAPTER 1	
Memory Overview	
Intel Memory Technologies	1-1
CHAPTER 2	
Flash Overview	
Flash Overview	2-1
APPLICATION NOTES	
AP-357 Power Supply Solutions for Flash Memory	2-7
AP-374 Flash Memory Write Protection Techniques	2-42
AB-29 Flash Memory Applications in Laser Printers	2-49
CHAPTER 3	
FlashFile™ Components	
DATA SHEETS	
DD28F032SA 32 Mbit (2 Mbit x 16, 4 Mbit x 8) FlashFile™ Memory	3-1
28F016SA 16 Mbit (1 Mbit x 16, 2 Mbit x 8) FlashFile™ Memory	3-6
28F008SA 8 Mbit (1 Mbit x 8) FlashFile™ Memory	3-49
28F008SA-L 8 Mbit (1 Mbit x 8) FlashFile™ Memory	3-77
APPLICATION NOTES	
AP-359 28F008SA Hardware Interfacing	3-105
AP-360 28F008SA Software Drivers	3-116
AP-362 Implementing Mobile PC Designs Using High Density FlashFile™ Components	3-139
AP-364 28F008SA Automation and Algorithms	3-194
AP-375 Upgrade Considerations from the 28F008SA to the 28F016SA	3-208
AP-377 28F016SA Software Drivers	3-221
AP-378 System Optimization Using the Enhanced Features of the 28F016SA	3-259
ENGINEERING REPORT	
ER-27 The Intel 28F008SA Flash Memory	3-282
SUPPORT TOOLS	
Intel 28F008SA FlashFile™ Memory Evaluation Module D, FLASHEVAL4 Product Brief	3-307
CHAPTER 4	
Boot Block Components	
DATA SHEETS	
28F400BX-T/B, 28F004BX-T/B 4 Mbit (256K x 16, 512K x 8) Boot Block Flash Memory Family	4-1
28F400BX-TL/BL, 28F004BX-TL/BL 4 Mbit (256K x 16, 512K x 8) Low Power Boot Block Flash Memory Family	4-49
28F200BX-T/B, 28F002BX-T/B 2 Mbit (128K x 16, 256K x 8) Boot Block Flash Memory Family	4-92
28F200BX-TL/BL, 28F002BX-TL/BL 2 Mbit (128K x 16, 256K x 8) Low Power Boot Block Flash Memory Family	4-138
28F001BX-T/28F001BX-B 1M (128K x 8) CMOS Flash Memory	4-179
APPLICATION NOTES	
AP-341 Designing an Updatable BIOS Using Flash Memory	4-208
AP-363 Extended Flash Bios Concepts for Portable Computers	4-251
ENGINEERING REPORTS	
ER-26 The Intel 28F001BX-T and 28F001BX-B Flash Memories	4-273
ER-29 The Intel 2/4-Mbit Boot Block Flash Memory Family	4-288
SUPPORT TOOLS	
Boot Block Flash: The Next Generation White Paper	4-317

Table of Contents (Continued)

Intel 2/4Mbit Boot Block Flash Memory Evaluation Module (D, FLASHEVAL5) Product Brief	4-321
CHAPTER 5	
Bulk-Erase Components	
DATA SHEETS	
28F020 2048K (256K x 8) CMOS Flash Memory	5-1
28F010 1024K (128K x 8) CMOS Flash Memory	5-33
28F512 512K (64K x 8) CMOS Flash Memory	5-64
28F256A 256K (32K x 8) CMOS Flash Memory	5-92
APPLICATION NOTES	
AP-316 Using Flash Memory for In-System Reprogrammable Nonvolatile Storage ..	5-117
AP-325 Guide to First Generation Flash Memory Programming	5-162
ENGINEERING REPORT	
ER-24 Intel Flash Memory 28F256A, 28F512, 28F010, 28F020	5-184
SUPPORT TOOLS	
Intel Flash Memory Evaluation Kit II (D, FLASHEVAL2) Product Brief	5-207
Small Outline Package Physical Dimensions	5-209
CHAPTER 6	
Memory Cards	
DATA SHEETS	
Series 2+ Flash Memory Cards, iMC040FLSP	6-1
Series 2+ Flash Memory Cards, iMC004FLSP/iMC020FLSP	6-6
Series 2 Flash Memory Cards, iMC002/004/010/020FLSA	6-37
iMC004FLKA 4-Megabyte Flash Memory Card	6-75
iMC002FLKA 2-Mbyte Flash Memory Card	6-105
iMC001FLKA 1-Megabyte Flash Memory Card	6-135
APPLICATION NOTES	
AP-343 Solutions for High Density Applications Using Intel Flash Memory	6-165
AP-361 Implementing the Integrated Registers of the Series 2 Flash Memory Card ..	6-195
AB-56 Preparing for the Next Generation Intel Flash Memory Card	6-212
SUPPORT TOOLS	
Intel FlashFile™ Memory-The Key to Diskless Mobile PCs	6-220
Intel ExCA™ Hardware Developer's Kit Product Brief	6-226
CHAPTER 7	
Flash ATA Drive	
Flash Drive iFD005P2SA/iFD010P2SA	7-1
iFD005P2SA/010P2SA Flash Drive Product Brief	7-32
CHAPTER 8	
Automotive Components	
A28F400BX-T/B 4-Mbit (256K x 16, 512K x 8) Boot Block Flash Memory Family (Automotive)	8-1
A28F200BX-T/B 2-Mbit (128K x 16, 256K x 8) Boot Block Flash Memory Family (Automotive)	8-35
A28F010 1024K (128 x 8) CMOS Flash Memory (Automotive)	8-68
A28F512 512K (64K x 8) CMOS Flash Memory (Automotive)	8-91
A28F256A 256K (32K x 8) CMOS Flash Memory (Automotive)	8-115
CHAPTER 9	
Process Engineering Reports	
ER-20 ETOX™ II Flash Memory Technology	9-1
ER-28 ETOX™ III Flash Memory Technology	9-6

Process Innovation	9-19
CHAPTER 10	
Article Reprints	
AR-710 Flash Solid-State Drive with 6MB/s Read/Write Channel and Data Compression	10-1
AR-711 Flash: Big News in Storage?	10-4
AR-715 Flash Memory: Meeting the Needs of Mobile Computing	10-8
AR-716 Flash Memory for Top Speeds in Mobile Computing Applications	10-16
AR-717 The Many Facets of Flash Memory	10-18
AR-718 Standardizing on a Flash File System	10-27
AR-723 Interfacing BootBlock Flash Memories to the MCS® 96 Family	10-31

(iii) Alphanumeric Index

28F001BX-T/28F001BX-B 1M (128K x 8) CMOS Flash Memory	4-179
28F008SA 8 Mbit (1 Mbit x 8) FlashFile™ Memory	3-49
28F008SA-L 8 Mbit (1 Mbit x 8) FlashFile™ Memory	3-77
28F010 1024K (128K x 8) CMOS Flash Memory	5-33
28F016SA 16 Mbit (1 Mbit x 16, 2 Mbit x 8) FlashFile™ Memory	3-6
28F020 2048K (256K x 8) CMOS Flash Memory	5-1
28F200BX-T/B, 28F002BX-T/B 2 Mbit (128K x 16, 256K x 8) Boot Block Flash Memory Family	4-92
28F200BX-TL/BL, 28F002BX-TL/BL 2 Mbit (128K x 16, 256K x 8) Low Power Boot Block Flash Memory Family	4-138
28F256A 256K (32K x 8) CMOS Flash Memory	5-92
28F400BX-T/B, 28F004BX-T/B 4 Mbit (256K x 16, 512K x 8) Boot Block Flash Memory Family	4-1
28F400BX-TL/BL, 28F004BX-TL/BL 4 Mbit (256K x 16, 512K x 8) Low Power Boot Block Flash Memory Family	4-49
28F512 512K (64K x 8) CMOS Flash Memory	5-64
A28F010 1024K (128 x 8) CMOS Flash Memory (Automotive)	8-68
A28F200BX-T/B 2-Mbit (128K x 16, 256K x 8) Boot Block Flash Memory Family (Automotive)	8-35
A28F256A 256K (32K x 8) CMOS Flash Memory (Automotive)	8-115
A28F400BX-T/B 4-Mbit (256K x 16, 512K x 8) Boot Block Flash Memory Family (Automotive)	8-1
A28F512 512K (64K x 8) CMOS Flash Memory (Automotive)	8-91
AB-29 Flash Memory Applications in Laser Printers	2-49
AB-56 Preparing for the Next Generation Intel Flash Memory Card	6-212
AP-316 Using Flash Memory for In-System Reprogrammable Nonvolatile Storage	5-117
AP-325 Guide to First Generation Flash Memory Programming	5-162
AP-341 Designing an Updatable BIOS Using Flash Memory	4-208
AP-343 Solutions for High Density Applications Using Intel Flash Memory	6-165
AP-357 Power Supply Solutions for Flash Memory	2-7
AP-359 28F008SA Hardware Interfacing	3-105
AP-360 28F008SA Software Drivers	3-116
AP-361 Implementing the Integrated Registers of the Series 2 Flash Memory Card	6-195
AP-362 Implementing Mobile PC Designs Using High Density FlashFile™ Components ...	3-139
AP-363 Extended Flash Bios Concepts for Portable Computers	4-251
AP-364 28F008SA Automation and Algorithms	3-194
AP-374 Flash Memory Write Protection Techniques	2-42
AP-375 Upgrade Considerations from the 28F008SA to the 28F016SA	3-208
AP-377 28F016SA Software Drivers	3-221
AP-378 System Optimization Using the Enhanced Features of the 28F016SA	3-259
AR-710 Flash Solid-State Drive with 6MB/s Read/Write Channel and Data Compression..	10-1
AR-711 Flash: Big News in Storage?	10-4
AR-715 Flash Memory: Meeting the Needs of Mobile Computing	10-8
AR-716 Flash Memory for Top Speeds in Mobile Computing Applications	10-16
AR-717 The Many Facets of Flash Memory	10-18
AR-718 Standardizing on a Flash File System	10-27
AR-723 Interfacing BootBlock Flash Memories to the MCS® 96 Family	10-31
Boot Block Flash: The Next Generation White Paper	4-317
DD28F032SA 32 Mbit (2 Mbit x 16, 4 Mbit x 8) FlashFile™ Memory	3-1
ER-20 ETOX™ II Flash Memory Technology	9-1
ER-24 Intel Flash Memory 28F256A, 28F512, 28F010, 28F020	5-184
ER-26 The Intel 28F001BX-T and 28F001BX-B Flash Memories	4-273
ER-27 The Intel 28F008SA Flash Memory	3-282
ER-28 ETOX™ III Flash Memory Technology	9-6

Alphanumeric Index (Continued)

ER-29 The Intel 2/4-Mbit Boot Block Flash Memory Family	4-288
ER-33 ETOX™ IV Flash Memory Technology: Insight to Intel's Fourth Generation Process Innovation	9-19
Flash Drive iFD005P2SA/iFD010P2SA	7-1
Flash Overview	2-1
iFD005P2SA/010P2SA Flash Drive Product Brief	7-32
iMC001FLKA 1-Megabyte Flash Memory Card	6-135
iMC002FLKA 2-Mbyte Flash Memory Card	6-105
iMC004FLKA 4-Megabyte Flash Memory Card	6-75
Intel 28F008SA FlashFile™ Memory Evaluation Module D, FLASHEVAL4 Product Brief...	3-307
Intel 2/4Mbit Boot Block Flash Memory Evaluation Module (D,FLASHEVAL5) Product Brief	4-321
Intel ExCA™ Hardware Developer's Kit Product Brief	6-226
Intel Flash Memory Evaluation Kit II (D, FLASHEVAL2) Product Brief	5-207
Intel FlashFile™ Memory-The Key to Diskless Mobile PCs	6-220
Intel Memory Technologies	1-1
Series 2 Flash Memory Cards, iMC002/004/010/020FLSA	6-37
Series 2+ Flash Memory Cards, iMC004FLSP/iMC020FLSP	6-6
Series 2+ Flash Memory Cards, iMC040FLSP	6-1
Small Outline Package Physical Dimensions	5-209

Most of this handbook is devoted to techniques and information to help you design and implement semiconductor memory in your application or system. In this section, however, the memory chip itself will be examined and the processing technology required to turn a bare slice of silicon into high performance memory devices is described. The discussion has been limited to the basics of MOS (Metal Oxide Semiconductor) technologies as they are responsible for the majority of memory devices manufactured at Intel.

There are three major MOS technology families—PMOS, NMOS, and CMOS (Figure 1). They refer to the channel type of the MOS transistors made with the technology. PMOS technologies implement p-channel transistors by diffusing p-type dopants (usually boron) into an n-type silicon substrate to form the source and drain. P-channel is so named because the channel is comprised of positively charged carriers. NMOS technologies are similar, but use n-type dopants (normally phosphorus or arsenic) to make n-channel transistors in p-type silicon substrates. N-channel is so named because the

channel is comprised of negatively charged carriers. CMOS or Complementary MOS technologies combine both p-channel and n-channel devices on the same silicon. Either p- or n-type silicon substrates can be used, however, deep areas of the opposite doping type (called wells) must be defined to allow fabrication of the complementary transistor type.

Most of the early semiconductor memory devices, like Intel's pioneering 1103 dynamic RAM and 1702 EPROM were made with PMOS technologies. As higher speeds and greater densities were needed, most new devices were implemented with NMOS. This was due to the inherently higher speed of n-channel charge carriers (electrons) in silicon along with improved process margins. CMOS technology has begun to see widespread commercial use in memory devices. It allows for very low power devices used for battery operated or battery back-up applications. Historically, CMOS has been slower than any NMOS device. Today, CMOS technology has been improved to produce higher speed devices.

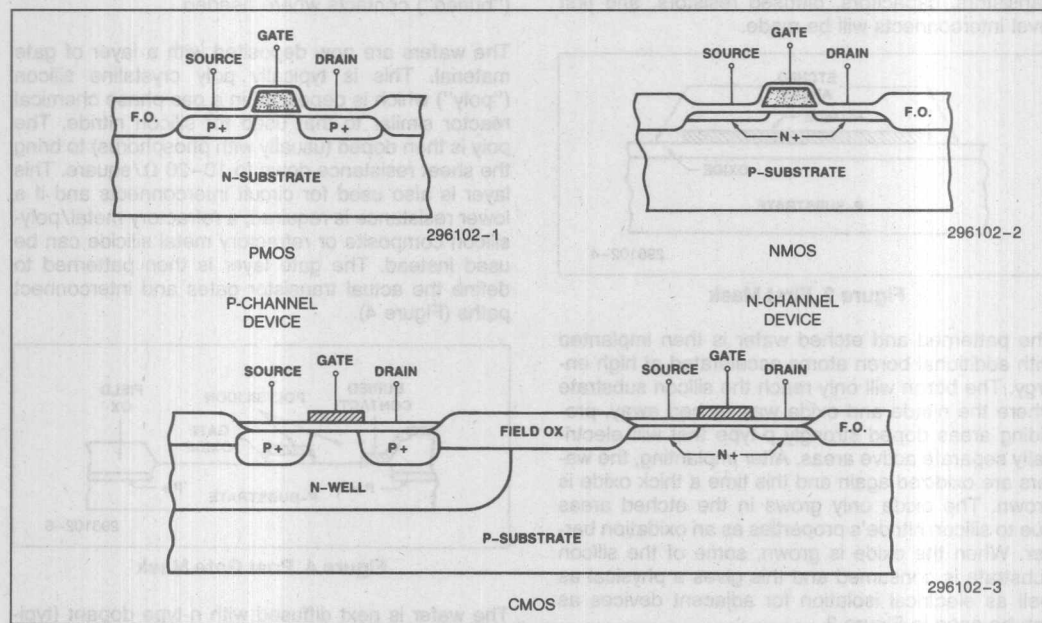


Figure 1. MOS Process Cross-sections

In the following section, the basic fabrication sequence for an HMOS circuit will be described. HMOS is a high performance n-channel MOS process developed by Intel for 5V single supply circuits. HMOS, and CHMOS, CHMOS-E (EPROM) and ETOX (Flash Memory), along with their evolutionary counterparts comprise the process family responsible for most of the memory components produced by Intel today.

The MOS IC fabrication process begins with a slice (or wafer) of single crystal silicon. Typically, it's 150 or 200 millimeter in diameter, about a half millimeter thick, and uniformly doped p-type. The wafer is then oxidized in a furnace at around 1000°C to grow a thin layer of silicon dioxide (SiO_2) on the surface. Silicon nitride is then deposited on the oxidized wafer in a gas phase chemical reactor. The wafer is now ready to receive the first pattern of what is to become a many layered complex circuit. The pattern is etched into the silicon nitride using a process known as photolithography, which will be described in a later section. This first pattern (Figure 2) defines the boundaries of the active regions of the IC, where transistors, capacitors, diffused resistors, and first level interconnects will be made.

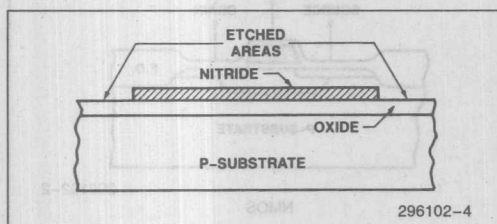


Figure 2. First Mask

The patterned and etched wafer is then implanted with additional boron atoms accelerated at high energy. The boron will only reach the silicon substrate where the nitride and oxide was etched away, providing areas doped strongly p-type that will electrically separate active areas. After implanting, the wafers are oxidized again and this time a thick oxide is grown. The oxide only grows in the etched areas due to silicon nitride's properties as an oxidation barrier. When the oxide is grown, some of the silicon substrate is consumed and this gives a physical as well as electrical isolation for adjacent devices as can be seen in Figure 3.

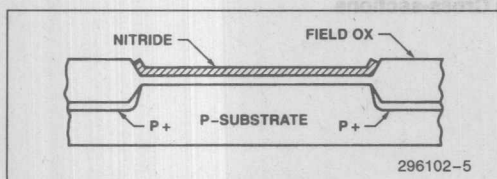


Figure 3. Post Field Oxidation

Having fulfilled its purpose, the remaining silicon nitride layer is removed. A light oxide etch follows taking with it the underlying first oxide but leaving the thick (field) oxide.

Now that the areas for active transistors have been defined and isolated, the transistor types needed can be determined. The wafer is again patterned and then if special characteristics (such as depletion mode operation) are required, it is implanted with dopant atoms. The energy and dose at which the dopant atoms are implanted determines much of the transistor's characteristics. The type of the dopant provides for depletion mode (n-type) or enhancement mode (p-type) operation.

The transistor types defined, the gate oxide of the active transistors are grown in a high temperature furnace. Special care must be taken to prevent contamination or inclusion of defects in the oxide and to ensure uniform consistent thickness. This is important to provide precise, reliable device characteristics. The gate oxide layer is then masked and holes are etched to provide for direct gate to diffusion ("buried") contacts where needed.

The wafers are now deposited with a layer of gate material. This is typically poly crystalline silicon ("poly") which is deposited in a gas phase chemical reactor similar to that used for silicon nitride. The poly is then doped (usually with phosphorus) to bring the sheet resistance down to 10-20 Ω /square. This layer is also used for circuit interconnects and if a lower resistance is required, a refractory metal/poly-silicon composite or refractory metal silicide can be used instead. The gate layer is then patterned to define the actual transistor gates and interconnect paths (Figure 4).

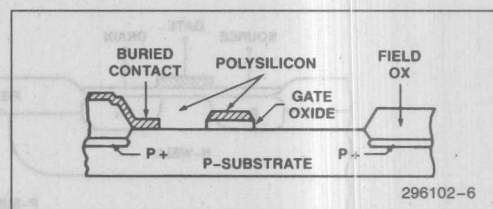


Figure 4. Post Gate Mask

The wafer is next diffused with n-type dopant (typically arsenic or phosphorus) to form the source and drain junctions. The transistor gate material acts as a barrier to the dopant providing an undiffused channel self-aligned to the two junctions. The wafer is then oxidized to seal the junctions from contamination with a layer of SiO_2 (Figure 5).

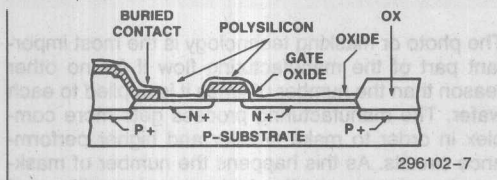


Figure 5. Post Oxidation

A thick layer glass is then deposited over the wafer to provide for insulation and sufficiently low capacitance between the underlying layers and the metal interconnect signals. (The lower the capacitance, the higher the inherent speed of the device.) The glass layer is then patterned with contact holes and placed in a high temperature furnace. This furnace step smooths the glass surface and rounds the contact edges to provide uniform metal coverage. Metal (usually aluminum or aluminum/silicon) is then deposited on the wafer and the interconnect patterns and external bonding pads are defined and etched (Figure 6). The wafers then receive a low temperature (approximately 500°C) alloy that insures good ohmic contact between the aluminum and diffusion or poly.

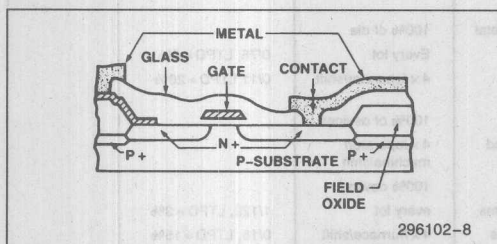


Figure 6. Complete Circuit (without passivation)

At this point the circuit is fully operational, however, the top metal layer is very soft and easily damaged by handling. The device is also susceptible to contamination or attack from moisture. To prevent this the wafers are sealed with a passivation layer of silicon nitride or a silicon and phosphorus oxide composite. Patterning is done for the last time opening up windows only over the bond pads where external connections will be made.

This completes basic fabrication sequence for a single poly layer process. Double poly processes such as those used for high density Dynamic RAMs, EPROMs, flash memories, and EEPROMs follow the same general process flow with the addition of gate, poly deposition, doping, and interlayer dielectric process modules required for the additional poly layer (Figure 7). These steps are performed right after the active areas have been defined (Figure 3) providing the capacitor or floating gate storage nodes on those devices.

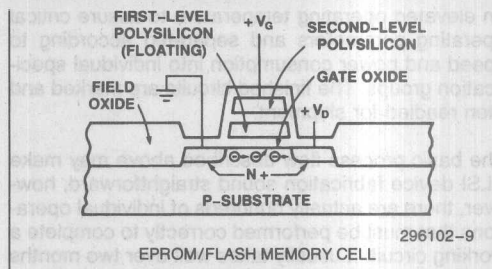


Figure 7. Double Poly Structure

After fabrication is complete, the wafers are sent for testing. Each circuit is tested individually under conditions designed to determine which circuits will operate properly both at low temperature and at conditions found in actual operation. Circuits that fail these tests are noted to distinguish them from good circuits. From here the wafers are sent for assembly where they are sawed into individual circuits with a paper-thin diamond blade. The noted circuits are then separated out and the good circuits are sent on for packaging.

Packages fall into two categories—hermetic and non-hermetic. Hermetic packages are Cerdip, where two ceramic halves are sealed with a glass frit, or ceramic with soldered metal lids. An example of hermetic package assembly is shown in Table 1. Non-hermetic packages are molded plastics.

The ceramic package has two parts, the base, which has the leads and die (or circuit) cavity, and the metal lid. The base is placed on a heater block and a metal alloy preform is inserted. The die is placed on top of the preform which bonds it to the package. Once attached, wires are bonded to the circuit and then connected to the leads. Finally the package is placed in a dry inert atmosphere and the lid is soldered on.

The cerdip package consists of a base, lead frame, and lid. The base is placed on a heater block and the lead frame placed on top. This sets the lead frame in glass attached to the base. The die is then attached and bonded to the leads. Finally the lid is placed on the package and it is inserted in a seal furnace where the glass on the two halves melt together making a hermetic package.

In a plastic package, the key component is the lead frame. The die is attached to a pad on the lead frame and bonded out to the leads with gold wires. The frame then goes to an injection molding machine and the package is formed around the lead frame. After mold the excess plastic is removed and the leads trimmed.

After assembly, the individual circuits are retested at an elevated operating temperature to assure critical operating parameters and separated according to speed and power consumption into individual specification groups. The finished circuits are marked and then readied for shipment.

The basic process flow described above may make VLSI device fabrication sound straightforward, however, there are actually hundreds of individual operations that must be performed correctly to complete a working circuit. It usually takes well over two months to complete all these operations and the many tests and measurements involved throughout the manufacturing process. Many of these details are responsible for ensuring the performance, quality, and reliability you expect from Intel products. The following sections will discuss the technology underlying each of the major process elements mentioned in the basic process flow.

PHOTOLITHOGRAPHY

The photo or masking technology is the most important part of the manufacturing flow if for no other reason than the number of times it is applied to each wafer. The manufacturing process gets more complex in order to make smaller and higher performance circuits. As this happens the number of masking steps increases, the features get smaller, and the tolerance required becomes tighter. This is largely because the minimum size of individual pattern elements determine the size of the whole circuit, effecting its cost and limiting its potential complexity. Early MOS IC's used minimum geometries (lines or spaces) of 8-10 microns (1 micron = 10^{-6} meter \approx 1/25,000 inch). The n-channel processes of the mid 1970's brought this down to approximately 5 microns, and today minimum geometries of 0.8 and even 0.6 microns are in production. This dra-

Table 1. Typical Hermetic Package Assembly

Flow	Process/Materials	Typical Item	Frequency	Criteria
	Wafer			
	Die saw, wafer break			
	Die wash and plate			
	Die visual inspection	Passivation, metal	100% of die	
	QA gate		Every lot	0/76, LTPD = 5%
	Die attach (Process monitor)	Wet out	4 x /operator/shift	0/11 LTPD = 20%
	Post die attach visual		100% of devices	
	Wire bond (Process monitor)	Orientation, lead dressing, etc.	4 x /operator/ machine/shift	
	Post bond inspection		100% devices	
	QA gate	All previous items	every lot	1/129, LTPD = 3%
	Seal and Mark (Process monitor)	Cap align, glass integrity, moisture	4 x /furnace/shift	0/15, LTPD = 15%
	Temp cycle		10 x to mil std. 883 cond. C	1/11, LTPD = 20%
	Hermeticity check (Process monitor)	F/G leak	100% devices	
	Lead Trim (Process monitor)	Burrs, etc. (visual) Fine leak	4 x /station/shift 2 x /station/shift	0/15, LTPD = 15% 1/129, LTPD = 3%
	External visual	Solder voids, cap alignment, etc.	100% devices	
	QA gate	All previous items	All lots	1/129, LTPD = 3%
	Class test (Process monitor)	Run standards (good and reject) Calibrate every system using "autover" program	Every 48 hrs.	
	Mark and Pack			
	Final QA	(See attached)		

NOTES:

1. Units for assembly reliability monitor.
2. Units for product reliability monitor.

296102-11

matic reduction in feature size was achieved using the newer high resolution photo resists and optimizing their processing to match improved optical printing systems.

A second major factor in determining the size of the circuit is the registration or overlay error. This is how accurately one pattern can be aligned to a previous one. Design rules require that space be left in all directions according to the overlay error so that unrelated patterns do not overlap or interfere with one another. As the error space increases the circuit size increases dramatically. Only a few years ago standard alignment tolerances were $\geq \pm 2$ microns; now advanced Intel processes have reduced this dramatically due mostly to the use of advanced projection and step and repeat exposure equipment.

The wafer that is ready for patterning must go through many individual steps before that pattern is complete. First the wafer is baked to remove moisture from its surface and is then treated with chemicals that ensure good resist adhesion. The thick photoresist liquid is then applied and the wafer is spun flat to give a uniform coating, critical for high resolution. The wafer is baked at a low temperature to solidify the resist into gel. It is then exposed with a machine that aligns a mask with the new pattern on it to a previously defined layer. The photo-resist will replicate this pattern on the wafer.

Negative working resists are polymerized by the light and the unexposed resist can be rinsed off with solvents. Positive working resists use photosensitive polymerization inhibitors that allow a chemically reactive developer to remove the exposed areas. The positive resists require much tighter control of exposure and development but yield higher resolution patterns than negative resistance systems.

The wafer is now ready to have its pattern etched. The etch procedure is specialized for each layer to be etched. Wet chemical etchants such as hydrofluoric acid for silicon oxide or phosphoric acid for aluminum are often used for this. The need for smaller features and tighter control of etched dimensions is increasing the use of plasma etching in fabrication. Here a reactor is run with a partial vacuum into which etchant gases are introduced and an electrical field is applied. This yields a reactive plasma which etches the required layer.

The wafer is now ready for the next process step. Its single journey through the masking process required the careful engineering of mechanics, optics, organic chemistry, inorganic chemistry, plasma chemistry, physics, and electronics.

DIFFUSION

The picture of clean room garbed operators tending furnace tubes glowing cherry red is the one most often associated with IC fabrication. These furnace operations are referred to collectively as diffusion because they employ the principle of solid state diffusion of matter to accomplish their results. In MOS processing, there are three main types of diffusion operations: predepos, drives, and oxidations.

Predeposition, or "predep," is an operation where a dopant is introduced into the furnace from a solid, liquid, or gaseous source and at the furnace temperature (usually 900°C–1200°C) a saturated solution is formed at the silicon surface. The temperature of the furnace, the dopant atom, and rate of introduction are all engineered to give a specific dose of the dopant on the wafer. Once this is completed the wafer is given a drive cycle where the dopant left at the surface by the predep is driven into the wafer by high temperatures. These are generally at different temperatures than the predepos and are designed to give the required junction depth and concentration profile.

Oxidation, the third category, is used at many steps of the process as was shown in the process flow. The temperature and oxidizing ambient can range from 800°C to 1200°C and from pure oxygen to mixtures of oxygen and other gases to steam depending on the type of oxide required. Gate oxides require high dielectric breakdown strength for thin layers (between 0.01 and 0.1 micron) and very tight control over thickness (typically ± 0.005 micron or less than $\pm 1/5,000,000$ inch), while isolation oxides need to be quite thick and because of this their dielectric breakdown strength per unit thickness is much less important.

The properties of the diffused junctions and oxides are key to the performance and reliability of the finished device so the diffusion operations must be extremely well controlled for accuracy, consistency and purity.

ION IMPLANT

Intel's high performance products require such high accuracy and repeatability of dopant control that even the high degree of control provided by diffusion operations is inadequate. However, this limitation has been overcome by replacing critical predepos with ion implantation. In ion implantation, ionized dopant atoms are accelerated by an electric field

1

and implanted directly into the wafer. The acceleration potential determines the depth to which the dopant is implanted.

The charged ions can be counted electrically during implantation giving very tight control over dose. The ion implanters used to perform this are a combination of high vacuum system, ion source, mass spectrometer, linear accelerator, ultra high resolution current integrator, and ion beam scanner. You can see that this important technique requires a host of sophisticated technologies to support it.

THIN FILMS

Thin film depositions make up most of the features on the completed circuit. They include the silicon nitride for defining isolation, polysilicon for the gate and interconnections, the glass for interlayer dielectric, metal for interconnection and external connections, and passivation layers. Thin film depositions are done by two main methods: physical deposition and chemical vapor deposition. Physical deposition is most common for deposition metal. Physical depositions are performed in a vacuum and are accomplished by vaporizing the metal with a high energy electron beam and redepositing it on the wafer or by sputtering it from a target to the wafer under an electric field.

Chemical vapor deposition can be done at atmospheric pressure or under a moderate vacuum. This type of deposition is performed when chemical gas-

es react at the wafer surface and deposit a solid film of the reaction product. These reactors, unlike their general industrial counterparts, must be controlled on a microscale to provide exact chemical and physical properties for thin films such as silicon dioxide, silicon nitride, and polysilicon.

The fabrication of modern memory devices is a long, complex process where each step must be monitored, measured and verified. Developing a totally new manufacturing process for each new product or even product line takes a long time and involves significant risk. Because of this, Intel has developed process families, such as HMOS, on which a wide variety of devices can be made. These families are scalable so that circuits need not be totally redesigned to meet your needs for higher performance.⁽¹⁾ They are evolutionary so that development time of new processes and products can be reduced without compromising Intel's commitment to consistency, quality, and reliability.

The manufacture of today's MOS memory devices requires a tremendous variety of technologies and manufacturing techniques, many more than could be mentioned here. Each requires a team of experts to design, optimize, control and maintain it. All these people and thousands of others involved in engineering, design, testing and production stand behind Intel's products.

(1) R. Pashley, K. Kokkonen, E. Boleky, R. Jecmen, S. Liu, and W. Owen, "H-MOS Scales Traditional Devices to Higher Performance Level," *Electronics*, August 18, 1977.

Flash Overview

2

2

The ideal memory system optimizes density, nonvolatility, fast readability and cost effectiveness. While traditional memory technologies may individually exhibit one or more of these desired characteristics, no single memory technology has achieved all of them without major tradeoffs—until the introduction of Intel Flash Memory.

WHAT IS FLASH MEMORY?

Introduced by Intel in 1988, ETOX flash memory is a high-density, truly nonvolatile and high performance read-write memory solution also characterized by low power consumption and extreme ruggedness and high reliability. The cost trend of Intel Flash Memory components continues to decline sharply due to: (1) manufacturing economies inherent in ETOX, Intel's industry-standard EPROM-based flash technology, (2) increases in memory density, and (3) rapid growth in production volume.

A comparison between Intel Flash Memory and other solid-state memory technologies underscores the fact that flash offers a design solution with distinct advantages. These advantages are key to future product differentiation for many applications requiring firmware updates or compact mass storage (Figure 1).

— ROM (read-only memory) is a mature, high density, nonvolatile, reliable and low cost memory technology widely used in PC and embedded applications. Once it is manufactured however, the contents of a ROM can never be altered. Additionally, initial ROM programming involves a time-consuming mask development process that requires stable code and is most cost-effective in high volumes.

Easy updatability makes flash memory clearly more flexible than ROM in most applications.

— SRAM (static random-access memory) is a high-speed, reprogrammable memory technology which is limited by its volatility and relatively low density. As a volatile memory technology, SRAM requires constant power to retain its contents. Built-in battery backup is therefore required when the main power source is turned off. Since battery failure is an inevitable fact of life, SRAM data loss is an ever-present danger. Additionally, SRAM requires four to six transistors to store one bit of information. This becomes a significant limitation in developing higher densities—effectively keeping SRAM cost relatively high.

Figure 1. Intel Flash Memory vs Traditional Memory Technologies

Memory	Inherently Non-Volatile	High Density	Low Power	One Transistor Cell	In-System Re-Writable	Code and Data Storage	Byte Alterable	Blocking	Hands off Updates
Flash	✓	✓	✓	✓	✓	✓	✓	✓	✓
SRAM + Battery					✓	✓	✓	✓	✓
DRAM + Disk		✓			✓	✓	✓	✓	✓
EEPROM	✓		✓		✓	✓	✓	✓	✓
OTP/EPROM	✓	✓	✓	✓					
Masked ROM	✓	✓	✓	✓					

*MS-DOS and Windows are registered trademarks of Microsoft Corporation.

November 1992

Order Number: 296101-004

In contrast, Intel flash memory is inherently nonvolatile, and the single transistor cell design of Intel's ETOX manufacturing process is extremely scalable, allowing the development of continuously higher densities and steady cost improvement over SRAM (Figure 2).

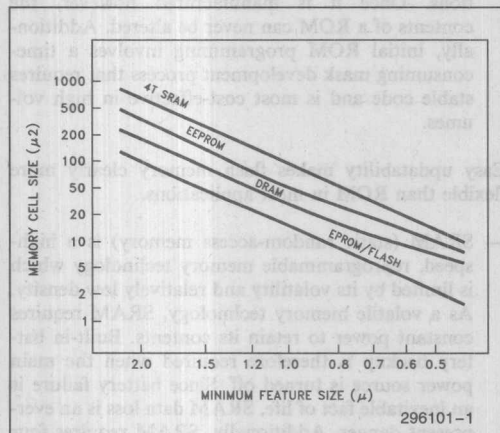


Figure 2

— EPROM (electrically programmable read-only memory) is a mature, high-density, nonvolatile technology which provides a degree of updatability not found in ROM. An OEM may program EPROM as needed to accommodate code changes or varying manufacturing unit quantities. Once programmed, however, the EPROM may only be erased by removing it from the system and then exposing the memory component to ultraviolet light—an impractical and time-consuming procedure for many OEMs and a virtually impossible task for end-users.

Unlike EPROM, flash memory is electrically re-writable within the host system, making it a much more flexible and easier to use alternative. Flash memory offers OEMs not only high density and nonvolatility, but higher functionality and the ability to differentiate their systems.

— EEPROM (electrically erasable programmable read-only memory) is nonvolatile and electrically byte-erasable. Such byte-alterability is needed in certain applications but involves a more complex cell structure, and significant trade-offs in terms of limited density, lower reliability and higher cost, making it unsuitable as a mainstream memory.

Unlike EEPROM, Intel flash memory technology utilizes a one-transistor cell, allowing higher densities, scalability, lower cost, and higher reliability, while taking advantage of in-system, electrical erasability (Figure 3).

	Intel ETOX Flash	EEPROM
Transistors	1	2
Cell Size (1-Micro Lithography)	15μ	38μ
Cycling Features	0.1%	5%

Figure 3

— DRAM (dynamic random access memory) is a volatile memory known for its density and low cost. Because of its volatility, however, it requires not only a constant power supply to retain data, but also an archival storage technology, such as disk, to back it up.

Partnered with hard disks for permanent mass storage, DRAM technology has provided a low-cost, yet space and power-hungry solution for today's PCs.

With ETOX process technology, Intel manufactures a flash memory cell that is 30% smaller than equivalent DRAM cells. Flash memory's scalability offers a price advantage as well, keeping price parity with DRAM, and also becoming more attractive as a hard disk replacement in portable systems as densities grow and costs decline.

Intel flash memory combines advantages from each of these memory technologies. In embedded memory applications, flash memory provides higher-performance and more flexibility than ROM and EPROM, while providing higher density and better cost effectiveness than battery-backed SRAM and EEPROM. Moreover, the true nonvolatility and low power consumption characteristics of flash memory make it a compelling alternative to DRAM in many applications.

ETOX III TECHNOLOGY

Unlike other approaches to flash memory, Intel ETOX is a proven technology. As its name suggests, ETOX (or "EPROM tunnel oxide") technology evolved from EPROM. With 95% process compatibility with EPROM, ETOX taps experience gained from a mature high-volume manufacturing base pioneered by Intel in the 1970s.

Data retention and lifetime reliability statistics for ETOX III flash are equivalent to those of EPROM. Representing the third generation of Intel flash memory technology, the ETOX III 0.8μ process provides <100 FITS (failures in time) @ 55°C in a specification that delivers 100,000 write cycles per block. This capability significantly exceeds the cycling requirements of even the most demanding applications.

For example, code storage for embedded control programs used in standard computer applications is infrequently updated. Twenty-year system lifetimes may require fewer than 100 rewrites. Even routinely-changed data tables (used in navigational computers and black box controllers) only require about 1,000 write cycles over a 20-year period. The most demanding flash memory application of all, archival data storage in PC applications, typically requires about 5,000 write cycles in 20 years.

ETOX flash memory's simple single-transistor cell structure makes it smaller than other flash cells, allowing designers to create highly integrated systems which are more reliable and cost-effective than those based on more complex and less mature flash technologies. The inherent scalability of ETOX III Flash Memory and high-volume manufacturing is enabling a corresponding downtrend in flash cost.

Flash memory has added a new dimension to nonvolatile memory applications. Embedded systems, such as PC BIOS, hard disk drive controllers and cellular telephone applications take advantage of the easy update capability, high density and high performance of Intel Flash Memory. Today's new generation of portable computers require the optimum combination of performance, size, weight, low power and shock resistance. Whether implemented in memory cards, solid state disks or at the component level, Intel's Flash Memories are also enabling a whole new generation of mobile computers.

IMPLEMENTING INTEL FLASH MEMORY

Today, Intel continues to serve both updatable nonvolatile memory applications as well as the rapidly emerging solid-state mass storage applications with flash memory solutions tailored to meet the needs of these markets.

Updatable Code Storage

Code and data storage comprise the updatable nonvolatile memory applications that require high performance, high density, and easy update capability. Because these applications are not updated as frequently as solid-state mass storage applications, erase/write cycles are not as critical as integration and performance requirements. This application segment is served effectively with full chip-erase or Boot Block products.

Intel's 28F001BX 1 Mbit Boot Block flash component, featuring a sectorized architecture, has been widely accepted in embedded code storage applications, particularly in PC BIOS and cellular communications. By adopting Boot Block for their products, over 20 PC manufacturers have gained added flexibility and the ability to differentiate in a highly competitive market. End users also benefit from the ability to upgrade BIOS software quickly and securely. The blocked architecture allows the OEM to store critical system code securely in the lockable "boot block" of the device that can minimally bring up the system and download to other locations of the device to initialize the system. The hardware boot locking feature guarantees that even if the power is disrupted during a BIOS update, the system will be able to recover immediately.

In response to customer requests for speed, density, low power, surface-mount options and an industry-standard upgrade path for portable computing and telecommunications, Intel more recently introduced the 2 Mbit 28F200BX and 4 Mbit 28F400BX Boot Block products.

These products offer 60 ns performance; two surface mount packages: 40-lead TSOP (X8 only) and 44-lead PSOP; and a proprietary Boot Block architecture similar to the 1 Mbit Boot Block device. The Boot Block stores the code necessary to initialize the system, while parameter blocks can be used to store manufacturing product code, setup parameters, and frequently updated code such as system diagnostics. The main operating code is stored in the main blocks. Both devices are available in a x16/x8 ROM-compatible pin-out in 44-lead PSOP surface mount packaging. These pinouts and packages allow an easy upgrade from 2 Mb to 4 Mb, since only one address is added to the 4 Mb device.

Solid-State Mass Storage

This major application segment requires very high density memory, automated programming and high-performance erase/write capability at a very low cost per bit. Erasing and writing portions of the code or data is much more frequent in solid-state mass storage than in updatable firmware applications.

Intel's symmetrically blocked 8 Mbit 28F008SA Flash-File™ memory is the highest density nonvolatile read/write solution for solid-state mass storage. What's more, it is the first flash memory device optimized for solid-state storage of software and data files.

The 28F008SA is packaged in an advanced 40-lead TSOP (thin, small outline package) or 44-lead PSOP (plastic SOP) to provide the extremely small form factor required for today's handheld, pen-based and sub-

*Based on 10 MB card design, 5,000 cycle yields 50,000 MB of stored data, which far exceeds most usage environments and file system methodologies.

notebook portable computers. The compactness of an 8 Mbit device in a TSOP package allows for high-density flash arrays to be included both on a system motherboard for direct execution of user programs or operating systems, as well as memory cards for transportable program and file storage.

Memory Cards

Intel's family of flash memory cards provides the most reliable and rugged form of removable memory media. High density, true nonvolatility, rewrite flexibility, and proven cost effectiveness make Intel Series 1 Flash Memory Cards the ideal medium for storing and updating application code as well as capturing data.

For file storage applications that require high performance, ruggedness, long battery life, small size and light weight, Intel's Series 2 Flash Memory Cards in 4-Mbyte, 10-Mbyte and 20-Mbyte densities provide the best solution. Based on Intel's 8-Mbit FlashFile memory components, the Series 2 card's block-erase functionality and high density take full advantage of flash filing systems like Microsoft's Flash File System software to provide full disk emulation in the form of removable, nonvolatile storage. The cards conform to the PCMCIA 2.0/JEIDA 4.1 68-pin standards and are compatible with Intel's Exchangeable Card Architecture (ExCA™) to ensure system-to-system interoperability.

THE IMPETUS BEHIND THE "SOLID-STATE" DISK

Because the disk-based PC is so prevalent and eminently familiar to both designers and end-users, many of today's portable systems still rely on it as their primary medium. At the same time, disk drive manufacturers have made great strides toward improving the reliability,

size and performance of their systems, as well as the disk media themselves.

Yet the disk drive is an electro-mechanical system with inherent limitations. Any mechanical hardware is much more vulnerable than solid state semiconductor technology to the shock, vibration, and impurities that portable PCs encounter during normal use. Hard disk drives can typically withstand up to 10Gs of operating shock; Intel FlashFile memory, with no moving parts, can withstand as much as 1000Gs. Additionally, Intel's Series 2 Memory Cards feature approximately 1.6 million hours mean time between failure (MTBF). Such endurance and reliability is essential for many of today's truly mobile handheld palmtop and notebook sized PCs, particularly within applications requiring extreme data integrity.

Power consumption is another major consideration for today's mobile PC designer and user. The drive typically requires anywhere from 3 watts to as many as 8 watts of power to run—which means rapid drain of the system's batteries. Compare this to flash memory in a hard disk configuration. It consumes less than one two-hundredth the average power of a comparable magnetic disk drive based on the typical user model. At the chip level, the 8 Mbit FlashFile Memory component has a DEEP POWERDOWN mode that reduces power consumption to less than 0.2 μ A.

Additional shortcomings of disk drives are their size, weight and floor costs. Magnetic drives do not scale well, that is, it becomes increasingly difficult to improve or even retain density as platter size shrinks. Thus, every reduction in drive size requires complete retooling and costly learning. Also, the complex controller circuitry provides a price floor under which magnetic drives cannot drop. Since flash is scalable, at some point in the near future, small magnetic drives are likely to become more expensive per Mbyte than flash cards and are certain to have less capacity.

	Disk/DRAM	Flash
Average Seek Time	28.0 ms	0
Latency	8.3 ms	0
Data Transfer Rate		
Read:	8 Mbits/sec.	106.7 Mbits/sec.
Write:	8 Mbits/sec.	1 Mbit/sec.
	... Now Read from RAM	Direct Processor Access
Total Time to Access (1 Kbyte File)	37.3 ms	0.077 ms

Figure 4

From a performance standpoint, disk-based systems still require some form of supplementary memory that is directly executable. Typically, DRAM is used for executable code storage and data manipulation. Data from the disk is downloadable into the DRAM cache before users can access the information. Then when a "save" operation is desired, the data is uploaded from DRAM back into the disk. This download/upload process slows down system throughput while the redundant memory media produce even more system overhead in the form of added space, power consumption and weight (Figure 4).

Today's PCs are typically configured with 4 Mbytes–8 Mbytes of DRAM backed up by at least a 40-Mbyte disk. FlashFile memory fully supports this system configuration when used simply as a magnetic drive replacement. Instructions and data are still swapped to DRAM, but at a faster rate. Plus execution speed can be enhanced if the DRAM is replaced with SRAM.

In the solid-state computer, the "DRAM + magnetic hard drive" is replaced by a "flash memory + SRAM". The key to this architecture is the ability to eXecute-In-Place (XIP). Program instructions stored in flash memory are read directly by the processor. Results are written directly to the flash memory. Compute-intensive operations that require the fastest memory access and byte-alterability can use high-speed SRAM or pseudo SRAM. Some of the system DRAM can be replaced by low-cost flash and a small part of the DRAM can be replaced by SRAM. The flash memory space is made even more storage efficient through the use of compression techniques which may offer up to 2:1 compression.

SOFTWARE DEVELOPMENTS POSITION FLASH FOR PORTABLE APPLICATIONS

The majority of today's portable computers and supporting software programmers are designed to run using Microsoft's MS-DOS* disk operating system. MS-DOS was developed to allow broad-based compatibility between systems and software and to optimize the sectoring scheme inherent to disk technology.

Intel's Flash Memory, based on a block-erase architecture, divides the flash memory into segments that are loosely analogous to the zones recognized in MS-DOS. Thanks to recent software developments, flash memory can effectively serve as the main memory within portable computers, providing user functions virtually identical to, and even improved over, those of disk-based systems.

Specifically, two recent developments allow this achievement: DOS in ROM-executable form (DOS was formerly designed to be stored on disk and then downloaded to/executed out of RAM); and a file system designed for flash memory technology that allows the devices to erase blocks of memory instead of the whole chip.

ROM-executable DOS provides several benefits to both system manufacturers and ultimately end users. First, since most of the operating system is composed of fixed code, the amount of system RAM required to execute DOS is reduced from 50K to 15K, thereby conserving system space and power. Secondly, DOS can now be permanently stored in and executed from a single ROM-type of device—flash memory—so systems come ready to run. Lastly, users enjoy "instant on" performance since the traditional disk-to-DRAM boot function and software downloading steps are eliminated.

For example, by storing application software and operating system code in a Resident Flash Array (RFA), users enjoy virtually instant-on performance and rapid in-place code execution. An RFA also protects against software obsolescence because it is in-system updatable. Resident software, stored in flash rather than disk, extends battery life and increases system reliability.

Since erasing and writing data to flash memory is a distinctly different operation than rewriting information to a disk, new software techniques were necessary to allow flash to emulate disk functionality. File management software like Microsoft's Flash File System (FFS) allows Intel's Flash Memory components and flash cards to emulate the file storage capabilities of disk. Microsoft's FFS transparently handles data swaps between flash blocks similar to the way MS-DOS handles swaps between disk sectors. Under FFS, the user may input a MS-DOS or Windows* command without regard for whether a flash memory or magnetic disk is installed in the system. The FFS also employs wear-leveling algorithms that prevent any block from being cycled excessively, thus assuring millions of hours of reliability. Flash filing systems make the management of flash memory devices completely transparent to the user.

CONCLUSION

Intel Flash Memory presents an entirely new memory technology alternative. As a high-density, nonvolatile read/write technology, it is exceptionally well-suited to serve as a solid-state disk or a cost-effective and highly reliable replacement for DRAMs and battery-backed static RAMs. Its inherent advantages over these technologies make it particularly useful in portable systems that require the utmost in low power, compact size, and ruggedness while maintaining high performance and full functionality.

Intel Flash Memory offers:

- **Inherent Nonvolatility:** Unlike static RAMs, no backup battery is required to ensure data retention. In contrast to DRAMs, flash requires no disk to provide backup storage of data, programs or files.
- **Cost-Effective High Density:** Today, Intel flash memories cost about the same as DRAMs and about one fourth of SRAMs on a per-bit basis. The broad acceptance of flash is driving manufacturing volumes up and costs down at an unprecedented rate, allowing flash to soon compete on a cost basis even with disk drive within the notebook, sub-notebook and palmtop markets.
- **Solid-State Performance:** Because it is a semiconductor technology, flash memory consumes much less power, is much lighter in weight and is smaller and more shock-resistant and reliable than disk drives. Mobile computers no longer have to drain the battery to run a disk drive motor or accommodate the disk assembly's added bulk and weight. Now users no longer have to be threatened with the possibility of losing their data after a disk crash when conditions become unusually rough.
- **Direct Execution:** Since no disk-to-DRAM download step, seek or latency times are incurred with

flash memory, users enjoy significantly higher speed program and file access, as well as systems that turn on instantly to wherever the user left off.

- **Easy Updatability:** Unlike other nonvolatile memory technologies—ROM that can never be altered after it is manufactured or EPROM that can only be erased by removing it from the system and exposing it to ultraviolet light—flash can be erased and reprogrammed electrically while resident in the host system.
- **Software Compatibility:** With Flash File System software and ROM-executable versions of the disk operating system (DOS), complete software compatibility between a user's desktop and portable system is ensured.
- **Exchangeable Card Architecture (ExCA™):** Through Intel's ExCA card interface standard, Intel's flash memory cards meet all specifications of PCMCIA 2.0, ensuring interchangeability between a host of PCMCIA-compatible systems.
- **Family of Products:** Intel Flash Memory products are currently available in component densities up to 8 Mbits, and in 4-Mbyte, 10-Mbyte and 20-Mbyte memory cards. Additionally, Intel offers Boot Block devices in densities up to 4 Mbits.

Since writing and writing data to flash memory is a distinctly different operation than writing information to a disk, new software techniques were necessary to allow flash to emulate disk functionality. The new agreement software like Microsoft's Flash File System (FFS) allows Intel's Flash Memory components and flash cards to emulate the file storage capabilities of disk. Microsoft's FFS transparently handles data exchange between flash blocks similar to the way MS-DOS handles data exchange between disk sectors. Under FFS, the user may input a MS-DOS or Windows command without regard for whether a flash memory or magnetic disk is installed in the system. The FFS also employs wear-leveling algorithms that prevent any one block of memory from being used excessively, thus ensuring millions of years of reliability. Flash filing systems make the management of flash memory devices completely transparent to the user.

CONCLUSION

Intel Flash Memory presents an entirely new memory technology alternative. As a high-density, nonvolatile read/write technology, it is exceptionally well-suited to serve as a solid-state disk or a cost-effective and reliable replacement for DRAMs and battery-backed static RAMs. Its inherent advantages over these technologies make it particularly useful in portable systems that require the utmost in low power, compact size and ruggedness while maintaining high performance and full functionality.

SOFTWARE DEVELOPMENTS POSITION FLASH FOR PORTABLE APPLICATIONS

The majority of today's portable computers and super-portable software environments are designed to run using Microsoft's MS-DOS™ disk operating system. MS-DOS was designed to allow broad-based compatibility between systems and software and to optimize the software systems inherent to disk technology.

Intel's Flash Memory, based on a block-erase architecture, divides the flash memory into segments that are closely analogous to the zones recognized in MS-DOS. Thanks to recent software developments, flash memory can effectively replace the main memory within portable computers, preserving user functions virtually identical to and even improved over those of disk-based systems.

Power Supply Solutions for Flash Memory

ANIL SAMA
BRIAN DIPERT
APPLICATIONS ENGINEERING
INTEL CORPORATION

September 1993

Power Supply Solutions for Flash Memory

CONTENTS	PAGE
1.0 INTRODUCTION	2-9
2.0 FLASH MEMORY POWER REQUIREMENTS	2-9
V_{CC} Characteristics	2-9
V_{PP} Characteristics	2-9
2.1 Supplies for Battery Powered Applications	2-10
2.2 Choice of a DC-DC Converter	2-10
The Modular Solution	2-10
The Discrete Switching Regulator Solution	2-10
Attributes of a DC-DC Converter ...	2-11
3.0 12V V_{PP} SOLUTIONS: CONVERTING UP FROM 5V	2-11
3.1 Maxim MAX732	2-11
3.2 Linear Technology LT1110-12	2-13
3.3 Linear Technology LT1109-12	2-14
3.4 Motorola MC34063A	2-16
4.0 12V V_{PP} SOLUTIONS: CONVERTING UP FROM 3V	2-17
4.1 Linear Technology LT1110-12	2-18
4.2 Maxim MAX732 @ 30 mA	2-19
4.3 Maxim MAX732 @ 60 mA	2-21

CONTENTS	PAGE
5.0 5V V_{CC} SOLUTIONS: CONVERTING UP FROM 3V	2-22
5.1 Maxim MAX658 @ 250 mA	2-23
5.2 Linear Tech LT1110-5 @ 150 mA	2-24
6.0 12V V_{PP} SOLUTIONS: DOWN-CONVERTING FROM A HIGHER VOLTAGE	2-25
6.1 Maxim MAX667	2-26
6.2 Linear Technology LT1111-5	2-27
6.3 National Semiconductor LM2940CT-12	2-28
7.0 12V V_{PP} FROM 12V UNREGULATED	2-29
8.0 SUMMARY	2-29
APPENDIX A: Modular Solutions	2-30
APPENDIX B: Survey of Solutions Presented	2-31
APPENDIX C: Sources for DC-DC Converters	2-32
APPENDIX D: Sources for Discrete Components	2-34
APPENDIX E: Other Design Considerations	2-36
APPENDIX F: PC Layouts	2-39

1.0 INTRODUCTION

Intel flash memory is rapidly being incorporated into a wide range of applications, adding enhanced capability to existing "traditional" memory markets, and creating new markets that exploit its benefits. Sometimes the design platforms may not possess the low powered 12V supply for writing flash memory. The system design engineer then needs to identify a power conversion solution with features and capabilities matching the needs of the application. For example, portable equipment requires a power supply converter that minimizes size and weight, maximizes efficiency to extend battery life, and can be switched into a standby mode to conserve power.

The following pages present some state of the art DC-DC converter solutions. These new solutions are smaller and more efficient than those typically seen in the past. Each of these solutions optimizes a subset of all possible power converter features. The choice of an optimal solution for a given application will be a tradeoff between several attributes. The solutions shown should meet the conversion needs of the majority of applications involving flash memory. Specifically, the solutions that follow encompass the following five categories:

- 5V to 12V conversion
- 3V (2 alkaline/NiCd cells) to 12V conversion
- 3V (2 Alkaline/NiCd cells) to 5V conversion
- Downconverting to 12V from a higher voltage
- Converting 12V unregulated to 12V regulated

More than one solution is presented within each of these categories. These different solutions have distinct optimal features/advantages. The optimal attributes of each solution are outlined. In addition, the appendix contains a survey of all solutions presented here, and provides a basis for comparing their features. The reader should reference it to choose an optimal solution for his/her application.

NOTE:

Solutions were selected from products offered by over thirty DC-DC converter vendors. Since this industry develops many new solutions each year, Intel recommends that designers contact vendors for latest products. Intel will continue to work with the industry to develop optimum solutions for power conversion. Intel Corporation assumes no responsibility for circuitry other than circuitry embodied in Intel products. No other circuit patent licenses are implied.

2.0 INTEL FLASH MEMORY POWER REQUIREMENTS

Intel flash memory is powered by two sources; a 5V V_{CC} line and a 12V V_{pp} line. V_{CC} is the primary power source and the only power source needed to read the memory. V_{pp} is required when writing or erasing the memory.

V_{CC} Characteristics

V_{CC} supplies power to the flash device during all operational modes. Maximum V_{CC} current is demanded by the device during the read operation. The data sheets for all Intel flash memory devices at the time this application note was written specify a maximum read current (I_{CC}) of 30 mA at $5V \pm 10\%$. This is the guaranteed worst case DC V_{CC} current that may be required by a flash device for reading one byte of data. If multiple components are read simultaneously, the V_{CC} current requirement increases proportionately. V_{CC} tolerance must be maintained to within specification limits at all times for proper functioning of the device.

V_{pp} Characteristics

The supplemental V_{pp} source provides the higher voltages needed to carry out the erase, erase verify, program, and program verify operations. Maximum V_{pp} current is typically demanded during the program and erase modes. Data sheets for all Intel flash memory devices at the time this application note was written specify a maximum I_{pp} current of 30 mA at $12V \pm 5\%$ for both program and erase operations. This is the guaranteed worst case V_{pp} supply current that will be required by a flash device for writing one byte of data or erasing one block/component. If multiple components are programmed/erased simultaneously, the current requirement increases proportionately. V_{pp} must be maintained to within specification limits at all times during device program, and erase. The tolerance specification on V_{pp} must be strictly maintained. Over-voltage can damage the device, and under-voltage can decrease specified device reliability. Although the 12V supply must meet these worst case specifications, power usage will typically be much lower. The lower typical values seen in the data sheets should be used in calculating typical battery life.

2.1 Supplies for Battery Powered Applications

In applications where batteries are the primary source of power, the power supplies providing V_{CC} and V_{pp} need to be selected very carefully. Optimized operating efficiency of these supplies is important to extend battery life. Another attractive feature is the capability of these supplies to be switched into a very low power shutdown mode. It is important to minimize this shutdown current consumption as well since flash memory V_{pp} generators will often be in this state for extended periods of time. Moreover, since these supplies are used in equipment that is physically small and space-constrained, size and height of the supply need to be minimized.

Where two alkaline/NiCd batteries are used as the primary source of power, the primary voltage varies depending on the type and the state of discharge of the batteries. For example, alkaline batteries start life off at 1.5V, but may still retain a significant amount of energy when the voltage falls to 1.0V with use, and will work all the way down to 0.8V. On the other hand, NiCd cells maintain a near constant voltage of 1.25V throughout most of their discharge cycle, and work down to 1.0V. A solution that derives V_{CC} or V_{pp} from 2 AA batteries must hence be capable of doing so from an input voltage that lies in the range of 1.6V to 3.0V.

It is best to directly convert the primary battery voltage into the various voltages needed throughout the system. A step conversion (e.g. a 3V to 5V converter for V_{CC} , followed by a 5V to 12V converter for V_{pp}) is not recommended, since the inefficiency involved in each conversion step combines into one large inefficiency for the sum 3V to 12V conversion. Section 4 presents appropriate 3V battery to 12V converter solutions. Most of the solutions presented in this application note, while specifically designed for battery powered applications, are also viewed as ideal for other applications that incorporate flash memory.

2.2 Choice of a DC-DC Converter

The solution to finding the right power supply for flash memory lies in picking the right DC-DC converter for the job. Two broad categories of DC-DC converters available in the market today can be applied towards this purpose. These are the low power hybrid DC-DC converter module (or modular solution), and the low power discrete switching regulator IC solution.

The Modular Solution

The modular solution generally consists of a push-pull (Royer type) oscillator built around an isolation transformer, and in some cases followed by a linear regulator; all of which is encapsulated within a module. This hybrid module includes all components that are required by the DC-DC converter, and so no additional design effort is needed. The input and output voltages are fixed, and the input and output are almost always isolated via the isolation transformer. The main advantage of these solutions is that no design effort and/or external components are involved. They simply plug into a socket on a PC board. Disadvantages include lower efficiency (generally 60%), larger size/height (in most cases), and higher cost (generally 3x to 10x the cost of discrete solutions).

It would seem that the integration inherent in these solutions contributes towards system reliability, however the type and quality of the discrete components used internal to these hybrid devices is open to question. The isolation offered between the input and output is viewed as overkill for flash applications, since the total power required is typically less than 1W. Note also that the isolation transformer is often the main reason for the lower efficiencies.

The Discrete Switching Regulator Solution

The discrete switching regulator IC solution consists of a DC-DC converter IC (containing a switching regulator controller and an output power switch), along with a few discrete external components (inductor, diode, capacitors, resistors, etc.). The layout of the power supply system in this case is mostly left up to the user. However, application notes and data sheets explain the design process, and provide recommended circuits for commonly used solutions. The design can be tailored to deliver different output voltages and current levels depending on the characteristics of the input voltage and the external components.

Some vendors offer fixed output voltage versions, further simplifying the design process. The newer generation of high frequency low power switching regulator ICs are specifically targeted at battery powered operation, and most can be switched into a low quiescent current shutdown mode to extend battery life. These have typical efficiencies in the 75% to 90% range. Furthermore, the higher switching frequencies of these new parts (typically 100 KHz to 200 KHz) allow the use of smaller external components, which are available in surface mount varieties. As a consequence, these newer solutions are overall much smaller than what was typically seen just a year ago.

Attributes of a DC-DC Converter

Several attributes of a power supply converter must be evaluated and prioritized when choosing the best solution for a given application. These attributes include:

- Input Voltage Range
- Output Voltage and Tolerance
- Output Current Capability
- Efficiency of Conversion
- Printed Circuit Area
- Height
- Total Cost
- Shutdown Capability
- Quiescent Current Consumed in Shutdown Mode
- Rise Time from Shutdown
- Surface Mountability

The reader is referred to Appendix B, which provides a survey of all the solutions that are presented in this application note, in order to compare their attributes.

This application note primarily presents state of the art discrete switching regulator IC solutions which have been carefully designed for operation with flash memory. Included along with schematics are component values and sources/contacts for obtaining all the components. Actual layouts have also been included where possible. These are provided in Appendix F.

NOTE:

External components recommended in the designs should be used. These components (inductors, capacitors, resistors) were chosen based on recommendations by the converter IC vendors and provide the necessary quality for a clean design. Alternate "equivalent" parts should be chosen with care as their resistive and inductive elements can affect the operation of the solution. Please contact the respective converter IC companies for assistance if you select an alternate value/source for passive components.

3.0 V_{PP} SOLUTIONS: CONVERTING UP FROM 5V

Most computer systems have available a 5V V_{CC} line that is used for the majority of system power. Frequently, this 5V supply is used to generate 12V for flash memory. This section presents some of the new state of the art solutions that can achieve this function. These are all discrete switching regulators that optimize different attributes, mentioned along with the main features section of each example. Refer to Appendix B for a more detailed comparison of the attributes of these solutions.

2

3.1 Maxim Integrated Products—MAX732: V_{PP} @ 30 mA, 60 mA, 120 mA

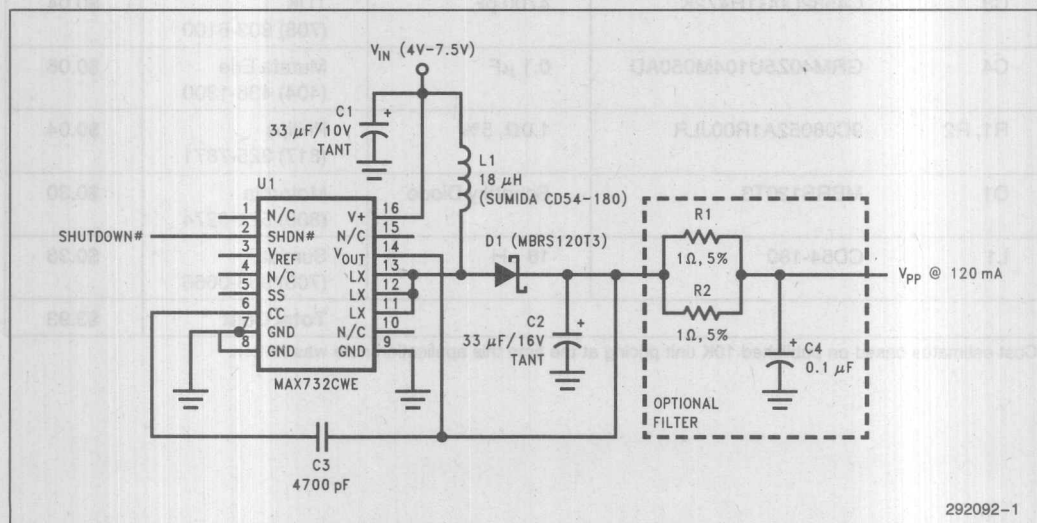


Figure 3-1. Maxim MAX732 5V to 12V Converter

Optimal Attributes

- Highest Efficiency
- Low Shutdown Current
- Wide Input Voltage Range
- All Surface Mount

Main Features

- Input Voltage Range: 4V to 7.5V
- Output Voltage: 12V \pm 4%
- Output Current Capability: Up to 120 mA
- Typical Efficiency: 90% at $I_{LOAD} = 60$ mA
- 170 KHz Operation
- Shutdown Feature On Chip
- Low Quiescent Current at Shutdown: 70 μ A typical
- Low Operating Quiescent Current: 1.6 mA typical
- Rise Time from Shutdown: 1 ms Typical
- Will Work off Existing 5V Supply or a 6 NiCd Battery Pack

The MAX732 design as shown is capable of providing up to 120 mA of V_{pp} current at an efficiency of 90%. The 5V input should be able to source the peak currents and start-up currents required by the circuit. This converter circuit can also run directly off a 6 cell NiCd pack present on many notebook/laptop computers. It is available in a 16-pin wide SOIC package, and uses small external surface mount components (5 in all). Voltage spikes may be present in the output due to incorrect layout, excessive output filter capacitor ESL (Equivalent Series Inductance) and diode switching transients. The optional filter circuit is recommended to eliminate any sharp transients. The supply can be switched into a shutdown mode where the output voltage falls to approximately $V_{CC} - 550$ mV. A layout is presented in Appendix F. Applications assistance and a surface mount evaluation kit is also available from Maxim.

Table 3-1. Parts List for the MAX732 5V to 12V Converter

Ref	Part #	Value/Type	Source	Cost*
U1	MAX732CWE	SMPS IC	Maxim (408) 737-7600	\$2.50
C1	267M1002-336-MR-720	33 μ F/10V Tantalum	Matsuo (714) 969-2491	\$0.31
C2	267M1602-336-MR-720	33 μ F/16V Tantalum	Matsuo (714) 969-2491	\$0.31
C3	C4532C0G1H472K	4700 pF	TDK (708) 803-6100	\$0.04
C4	GRM40Z5U104M050AD	0.1 μ F	Murata Erie (404) 436-1300	\$0.06
R1, R2	9C08052A1R00JLR	1.0 Ω , 5%	Philips (817) 325-7871	\$0.04
D1	MBRS120T3	Schottky Diode	Motorola (800) 521-6274	\$0.30
L1	CD54-180	18 μ H	Sumida (708) 956-0666	\$0.38
Total Cost				\$3.93

*Cost estimates based on published 10K unit pricing at the time this application note was written.

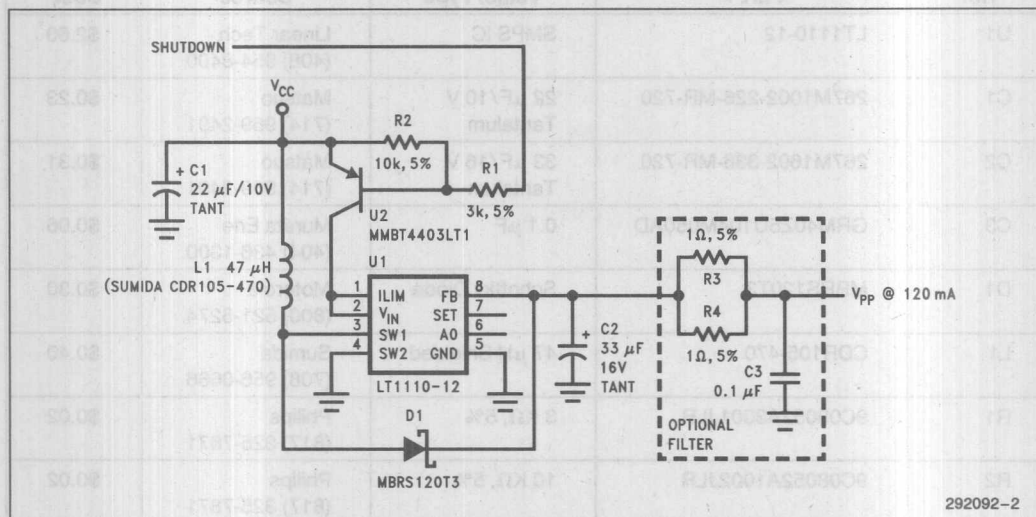


Figure 3-2. Linear Technology LT1110-12 5V to 12V Converter

Optimal Attributes

- Small Size: 0.45 sq. in. Total Board Area (Single Sided)
- Very Low Shutdown Current: 16 μ A
- All Surface Mount

Main Features

- Input Voltage Range: 4.5V to 5.5V
- Output Voltage: 12V \pm 5%
- Output Current Capability: Up to 120 mA
- Typical Efficiency: 76%
- 60 KHz Operation
- Shutdown Possible Using External Components as Shown
- Low Quiescent Current at Shutdown: 16 μ A typical
- Rise Time from shutdown: 800 μ s typical

The Linear Technology LT1110-12 is a fixed 12V output part which is well suited to flash memory applications. The part is available in a small 8-pin surface mount (SO8) package. The part needs 7 external components to implement a small size 5V to 12V converter solution that can be shutdown to a very low quiescent current state—16 μ A typical. The 5V source must be capable of supplying the instantaneous start-up and peak currents required during operation. Voltage spikes may be present in the output due to incorrect layout, excessive output filter capacitor ESL (Equivalent Series Inductance) and diode switching transients. The optional RC filter circuit is recommended in order to eliminate these sharp transients. The output voltage during shutdown falls to approximately $V_{IN} - 550$ mV. A recommended board layout appears in Appendix F. Applications assistance is available from Linear Technology Corporation.

Table 3-2. Part List for the LT1110-12 5V to 12V Converter

Ref	Part #	Value/Type	Source	Cost*
U1	LT1110-12	SMPS IC	Linear Tech (408) 954-8400	\$2.60
C1	267M1002-226-MR-720	22 μ F/10 V Tantalum	Matsuo (714) 969-2491	\$0.23
C2	267M1602-336-MR-720	33 μ F/16 V Tantalum	Matsuo (714) 969-2491	\$0.31
C3	GRM40Z5U104M050AD	0.1 μ F	Murata Erie (404) 436-1300	\$0.06
D1	MBRS120T3	Schottky Diode	Motorola (800) 521-6274	\$0.30
L1	CDR105-470	47 μ H Shielded	Sumida (708) 956-0666	\$0.40
R1	9C08052A3001JLR	3 K Ω , 5%	Philips (817) 325-7871	\$0.02
R2	9C08052A1002JLR	10 K Ω , 5%	Philips (817) 325-7871	\$0.02
R3, R4	9C08052A1R00JLR	1 Ω , 5%	Philips (817) 325-7871	\$0.04
U2	MMBT4403LT1	2N4403 PNP Transistor	Motorola (800) 521-6274	\$0.09
Total Cost				\$4.07

*Cost estimates based on published 10K unit pricing at the time this application note was written.

3.3 Linear Technology LT1109-12: V_{PP} @ 30 mA, 60 mA

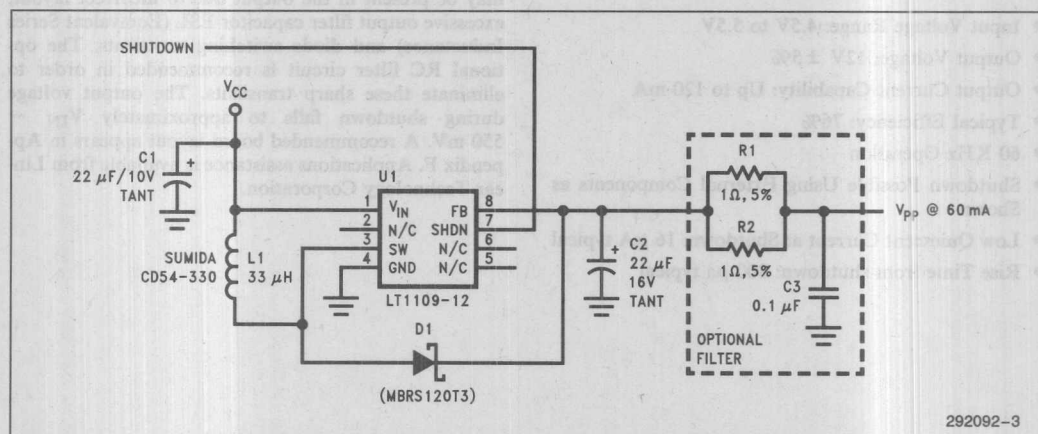


Figure 3-3. Linear Technology LT1109-12 5V to 12V Converter

Optimal Attributes

- Smallest Size
- Low Shutdown Current
- All Surface Mount

Main Features

- Input Voltage Range: 4.5V to 5.5V
- Output Voltage: 12V \pm 5%
- Output Current Capability: Up to 60 mA
- Typical Efficiency: 84%
- 130 KHz Operation
- Shutdown Feature On Chip
- Low Quiescent Current at Shutdown: 375 μ A typical
- Rise Time from shutdown: 800 μ s typical
- Small Size: SO8 plus 4 small external components

The Linear Technology LT1109-12 is a fixed 12V output part which is very well suited to flash memory applications. The part is available in a very small 8-pin surface mount (SO8) package. The part needs just 4 small external components to implement an extremely small size 5V to 12V converter solution that can be shutdown to a low quiescent current state—375 μ A typical. The 5V source must be capable of supplying the instantaneous start-up and peak currents required by the operation. Voltage spikes may be present in the output due to incorrect layout, excessive output filter capacitor ESL (Equivalent Series Inductance) and diode switching transients. The optional RC filter circuit is recommended in order to eliminate these sharp transients. The output during shutdown falls to approximately $V_{IN} - 550$ mV. A typical board layout is presented in Appendix F. Applications assistance is available from Linear Technology Corporation.

2

Table 3-3. Parts List for the LT1109-12 5V to 12V Converter

Ref	Part #	Value/Type	Source	Cost*
U1	LT1109-12	SMPS IC	Linear Tech (408) 432-1900	\$2.37
C1	267M1002-226-MR-720	22 μ F/10V Tant Chip Capacitor	Matsuo (714) 969-2491	\$0.23
C2	267M2502-106-MR-720	10 μ F/25V Tant Chip Capacitor	Matsuo (714) 969-2491	\$0.29
C3	GRM40Z5U104M050AD	0.1 μ F	Murata Erie (404) 436-1300	\$0.06
D1	MBRS120T3	Schottky Diode	Motorola (800) 521-6274	\$0.30
R1, R2	9C08052A1R00JLR	1 Ω , 5%	Philips (817) 325-7871	\$0.04
L1	CD54-330	33 μ H	Sumida (708) 956-0666	\$0.32
Total Cost				\$3.61

*Cost estimates based on published 10K unit pricing at the time this application note was written.

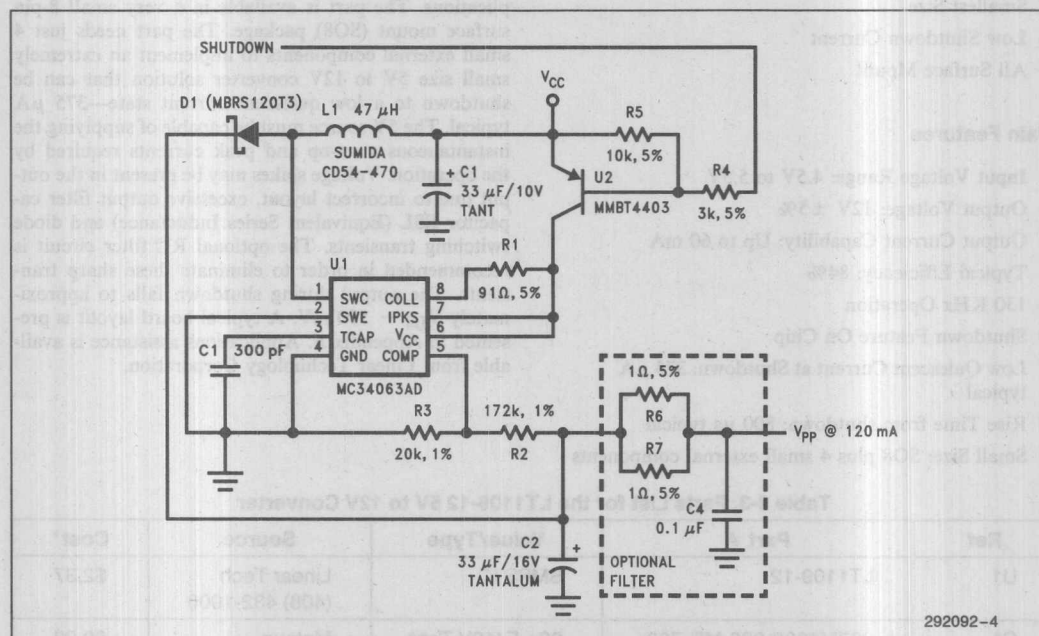
3.4 Motorola MC34063A: V_{pp} @ 30 mA, 60 mA, 120 mA

Figure 3-4. Motorola MC34063A 5V to 12V Converter

Optimal Attributes

- Lowest Cost
- Very Low Shutdown Current
- All Surface Mount

Main Features

- Input Voltage Range: 4.5V to 5.5V
- Output Voltage: 12V \pm 5%
- Output Current Capability: Up to 120 mA
- Typical Efficiency: 80%
- 100 KHz Operation
- Shutdown Feature Using External Components
- Low Quiescent Current at Shutdown: 25 μ A typical
- Rise Time From Shutdown: 2 ms typical
- SO8 Plus 11 Small External Components—All SMD

The Motorola MC34063A solution presented uses 11 small sized external components to implement a low cost surface mount 5V to 12V converter solution. Three external components (U2, R4, R5) are used to shut down supply to the part when V_{pp} is not needed. These could be eliminated to further lower the cost if power consumption is not important. The quiescent current in shutdown state is a low 25 μ A. The output voltage in shutdown is $V_{CC} - 550$ mV. Voltage spikes may be present in the output due to incorrect layout, excessive output filter capacitor ESL (Equivalent Series Inductance) and diode switching transients. The optional RC filter circuit is recommended in order to eliminate these sharp transients. Applications assistance is available from Motorola.

Table 3-4. Parts List for the MC34063A 5V to 12V Converter

Ref	Part #	Value/Type	Source	Cost*
U1	MC34063AD	SMPS IC (SO8)	Motorola (800) 521-6274	\$0.63
R1	9C08052A9100JLR	91 Ω , 5%	(Philips (817) 325-7871	\$0.02
R2	9B08053A1723FCB	172 K Ω , 1%	(Philips (817) 325-7871	\$0.04
R3	9B08053A2002FCB	20 K Ω , 1%	Philips (817) 325-7871	\$0.04
R4	9C08052A3001JLR	3 K Ω , 5%	Philips (817) 325-7871	\$0.02
R5	9C08052A1002JLR	10 K Ω , 5%	Philips (817) 325-7871	\$0.02
R6, R7	9C08052A1R00JLR	1 Ω , 5%	Philips (817) 325-7871	\$0.04
C1	267M1002-336-MR-720	33 μ F/16V Tantalum	Matsuo (714) 969-2491	\$0.28
C2	267M1602-336-MR-720	33 μ F/16V Tantalum	Matsuo (714) 969-2491	\$0.31
C3	GRM40X7R301M050AD	300 pF	Murata Erie (404) 436-1300	\$0.03
C4	GRM40Z5U104M050AD	0.1 μ F	Murata Erie (404) 436-1300	\$0.06
D1	MBRS120T3	Schottky Diode	Motorola (800) 521-6274	\$0.30
L1	CD54-470	47 μ H	Sumida (708) 956-0666	\$0.37
U2	MMBT4403LT1	PNP Transistor	Motorola (800) 521-6274	\$0.09
Total Cost				\$2.25

* Cost estimates based on published 10K unit pricing at the time this application note was written.

4.0 V_{pp} SOLUTIONS: CONVERTING UP FROM 2 NiCd/ALKALINE CELLS

Palmtop computers that use 2 alkaline/NiCd batteries require that the system work even when the battery

voltage is down near 1.8V. Currently there exist two good solutions that achieve a 12V output with inputs as low as 1.8V, and yet supply at least 30 mA of current. These are the LT1110-12 from Linear Technology Corporation, and the MAX732 from Maxim Integrated Products.

4.1 Linear Technology LT1110-12: V_{PP} @ 30 mA from 2 AA Cells

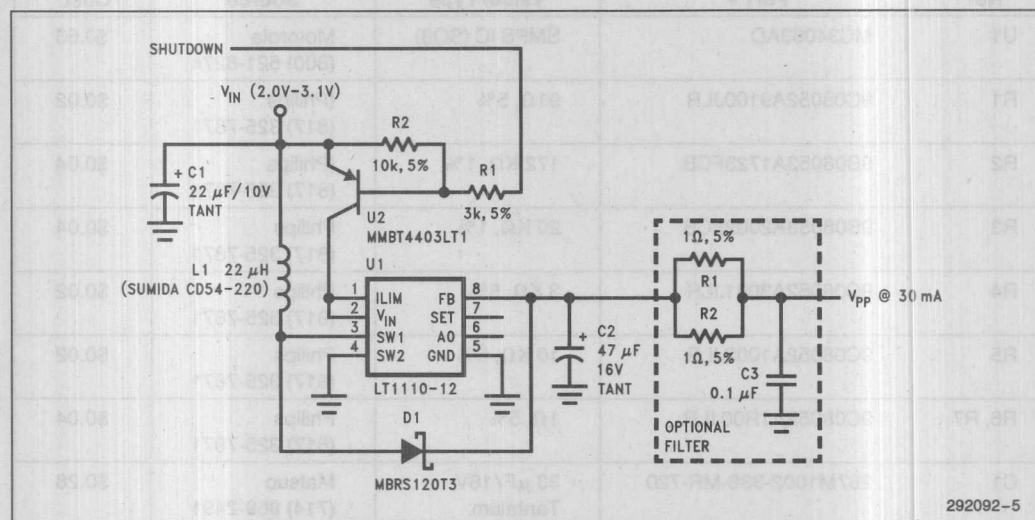


Figure 4-1. Linear Technology LT1110-12 3V to 12V Converter

Optimal Attributes

- Smallest Size
- Low Shutdown Current
- All Surface Mount

Main Features

- Input Voltage Range: 2.0V to 3.1V
- Output Voltage: 12V $\pm 5\%$
- Output Current Capability: Up to 30 mA
- Typical Efficiency: 70%
- 60 KHz Operation
- Shutdown Mode Using External Components
- Low Quiescent Current at Shutdown: 16 μA typical
- Rise Time from Shutdown: 4 ms typical

The LT1110-12 from Linear Technology Corporation, as shown, can be used to generate V_{PP} from an input voltage between 2.0V and 3.1V (most of the usable life of 2 alkaline/NiCd cells in series). This design is similar to the 5V to 12V converter design presented in Section 3.2. Replacing L1 and C2 with a lower inductance and a higher capacitance, respectively, allows the part to work down to 2.0V, while reducing the output current capability to 30 mA. The external PNP transistor is used to shut off the input supply to the converter IC, and puts the part in shutdown state. Note that a disadvantage of this scheme of shutdown is that the control signal source sinks approximately 5 mA ($V_{CC}/1K$) when the part is not in shutdown. However, the quiescent current in shutdown state is a low 16 μA . See Appendix E for an alternate shutdown solution. The output voltage in shutdown falls to approximately $V_{IN} - 550$ mV. Voltage spikes may be present in the output due to incorrect layout, excessive output filter capacitor ESL (Equivalent Series Inductance) and diode switching transients. The optional RC filter circuit is recommended in order to eliminate any sharp transients. A surface mount layout appears in Appendix F.

Optimal Attributes

- Highest Efficiency
- All Surface Mount

Main Features

- Input Voltage Range: 1.8V to 5.0V
- Output Voltage: 12V \pm 4%
- Output Current Capability: Up to 30 mA
- Typical Efficiency: 87%
- 170 KHz Operation
- Shutdown Mode On Chip
- Low Quiescent Current at Shutdown: 45 μ A typical
- Rise Time from shutdown: 25 ms typical

The MAX732 circuit as shown here can provide up to 30 mA at 12V from an input voltage as low as 1.8V. Note that the chip itself is powered from the 5V V_{CC} line required to use present day flash memory devices, whereas the inductor is connected to the primary battery supply. Voltage spikes may be present in the output due to incorrect layout, excessive output filter capacitor ESL and diode switching transients. The optional RC filter circuit is recommended in order to eliminate these sharp transients. Applications assistance and an evaluation kit is available from Maxim.

Table 4-2. Parts List for the MAX732 3V to 12V Converter (30 mA)

Ref	Part #	Value/Type	Source	Cost*
U1	MAX732CWE	SMPS IC	Maxim (408) 737-7600	\$2.50
C1, C2	267M1002-336-MR-720	33 μ F/10V Tantalum	Matsuo (714) 969-2491	\$0.56
C3	267M1002-475-MR-720	4.7 μ F/10V Tantalum	Matsuo (714) 969-2491	\$0.20
C4	GRM40X7R473M050AD	47 nF	Murata Erie (404) 436-1300	\$0.08
C5, C8	GRM40Z5U104M050AD	0.1 μ F	Murata Erie (404) 436-1300	\$0.12
C6, C7	267M1602-336-MR-720	33 μ F/16V Tantalum	Matsuo (714) 969-2491	\$0.62
R1, R2	9C08052A1R00JLR	1 Ω , 5%	Philips (817) 325-7871	\$0.04
D1	MBRS120T3	Schottky Diode	Motorola (800) 521-6274	\$0.30
L1	CD54-330	33 μ H	Sumida (708) 956-0666	\$0.38
Total Cost				\$4.80

*Cost estimates based on published 10K unit pricing at the time this application note was written.

4.3 Maxim Integrated Products—MAX732: V_{PP} @ 60 mA

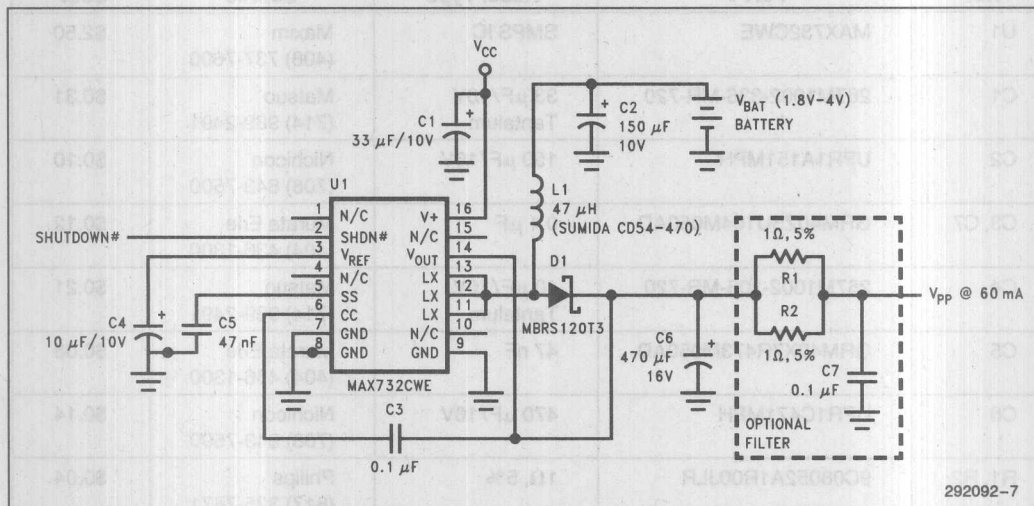


Figure 4-3. Maxim MAX732 3V to 12V Converter (60 mA)

Optimal Attributes

- Highest Efficiency
- 60 mA Output Current Capability

Main Features

- Input Voltage Range: 1.8V to 5.0V
- Output Voltage: 12V \pm 4%
- Output Current Capability: Up to 60 mA
- Typical Efficiency: 87%
- 170 KHz Operation
- Shutdown Mode On Chip
- Low Quiescent Current at Shutdown: 45 μ A typical
- Rise Time from shutdown: 75 ms typical

The MAX732 circuit as shown here can provide up to 60 mA at 12V from an input voltage as low as 1.8V. This solution is similar to the previous one presented but is not entirely surface mountable, because of the larger output and input filter capacitors. Currently, it is the only solution employing a single IC that can provide 60 mA at 12V from a 1.8V input. The 470 μ F/16V filter capacitor must be a low-ESR (Equivalent Series Resistance) type. Voltage spikes may be present in the output due to incorrect layout, excessive output filter capacitor ESL (Equivalent Series Inductance) and diode switching transients. The optional RC filter circuit is recommended in order to eliminate these sharp transients. Applications assistance and an evaluation kit is available from Maxim.

Table 4-3. Parts List for the MAX732 3V to 12V Converter (60 mA)

Ref	Part #	Value/Type	Source	Cost*
U1	MAX732CWE	SMPS IC	Maxim (408) 737-7600	\$2.50
C1	267M1002-336-MR-720	33 μ F/10V Tantalum	Matsuo (714) 969-2491	\$0.31
C2	UPR1A151MPH	150 μ F/10V	Nichicon (708) 843-7500	\$0.10
C3, C7	GRM40Z5U104M050AD	0.1 μ F	Murata Erie (404) 436-1300	\$0.12
C4	267M1002-106-MR-720	10 μ F/10V Tantalum	Matsuo (714) 969-2491	\$0.21
C5	GRM40X7R473M050AD	47 nF	Murata Erie (404) 436-1300	\$0.08
C6	UPR1C471MPH	470 μ F/16V	Nichicon (708) 843-7500	\$0.14
R1, R2	9C08052A1R00JLR	1 Ω , 5%	Philips (817) 325-7871	\$0.04
D1	MBRS120T3	Schottky Diode	Motorola (800) 521-6274	\$0.30
L1	CD75-470	47 μ H	Sumida (708) 956-0666	\$0.38
Total Cost				\$4.15

*Cost estimates based on published 10K unit pricing at the time this application note was written.

5.0 V_{CC} SOLUTIONS: CONVERTING UP FROM TWO NiCd/ALKALINE CELLS

Palmtop and hand-held computers that use two AA size NiCd or alkaline batteries need a converter solu-

tion to provide the V_{CC} supply for the system as well as flash memory. Two good solutions are offered currently for this purpose, the MAX658 from Maxim Integrated Products and the LT1110-5 from Linear Technology Corporation.

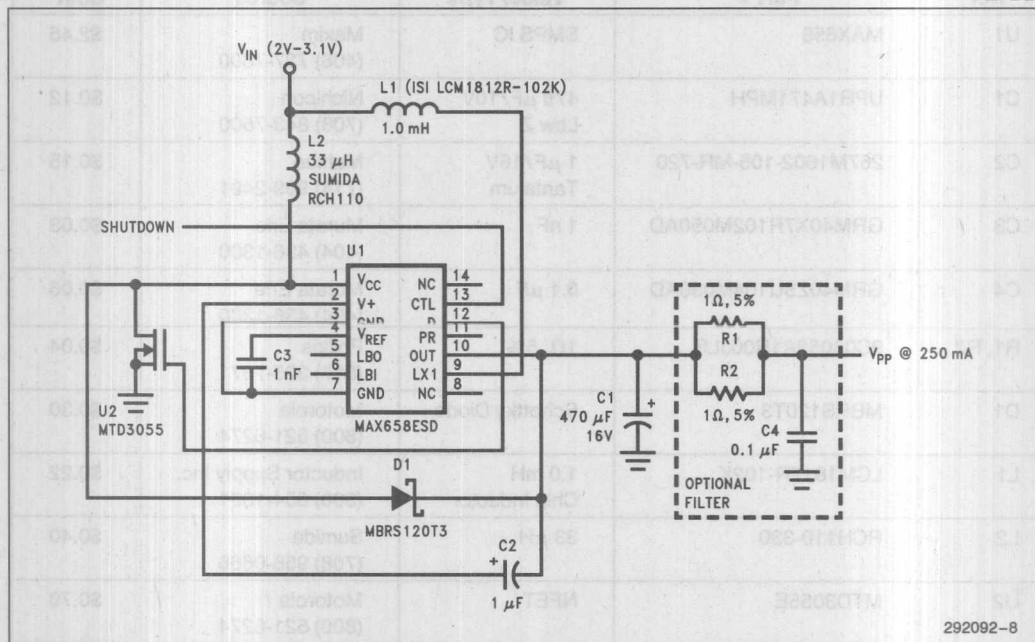


Figure 5-1. Maxim MAX658 3V to 5V Converter (250 mA)

Optimal Attributes

- Highest Efficiency
- 250 mA Output Current Capability
- Low Shutdown Current

Main Features

- Input Voltage Range: 2.0V to 3.1V
- Output Voltage: 5V \pm 10%
- Output Current Capability: Up to 250 mA
- Typical Efficiency: 85%
- 18 KHz Operation
- Shutdown Mode On Chip
- Low Quiescent Current at Shutdown: 80 μ A typical
- Rise Time from shutdown: 25 ms typical

The MAX658, available from Maxim Integrated Products in a 14-pin surface mount package, is a good high current solution for obtaining V_{CC} from a pair of NiCd/alkaline cells. The entire solution, however, is not 100% surface mountable. It uses a high current through-hole inductor and a large through-hole filter capacitor at the output. Voltage spikes may be present in the output due to incorrect layout, excessive output filter capacitor ESR (Equivalent Series Resistance) and diode switching transients. The optional RC filter circuit is recommended in order to eliminate any sharp transients. Applications assistance and an evaluation kit are available from Maxim.

Table 5-1. Parts List for the MAX658 3V to 5V Converter

Ref	Part #	Value/Type	Source	Cost*
U1	MAX658	SMPS IC	Maxim (408) 737-7600	\$2.45
C1	UPR1A471MPH	470 μ F/10V Low Z	Nichicon (708) 843-7500	\$0.12
C2	267M1602-105-MR-720	1 μ F/16V Tantalum	Matsuo (714) 969-2491	\$0.15
C3	GRM40X7R102M050AD	1 nF	Murata Erie (404) 436-1300	\$0.03
C4	GRM40Z5U104M050AD	0.1 μ F	Murata Erie (404) 436-1300	\$0.06
R1, R2	9C08052A1R00JLR	1 Ω , 5%	Philips (817) 325-7871	\$0.04
D1	MBRS120T3	Schottky Diode	Motorola (800) 521-6274	\$0.30
L1	LCM1812R-102K	1.0 mH Chip Inductor	Inductor Supply Inc. (800) 854-1881	\$0.22
L2	RCH110-330	33 μ H	Sumida (708) 956-0666	\$0.40
U2	MTD3055E	NFET	Motorola (800) 521-6274	\$0.70
Total Cost				\$4.47

*Cost estimates based on published 10K unit pricing at the time this application note was written.

5.2 Linear Technology LT1110-5: V_{CC} @ 150 mA

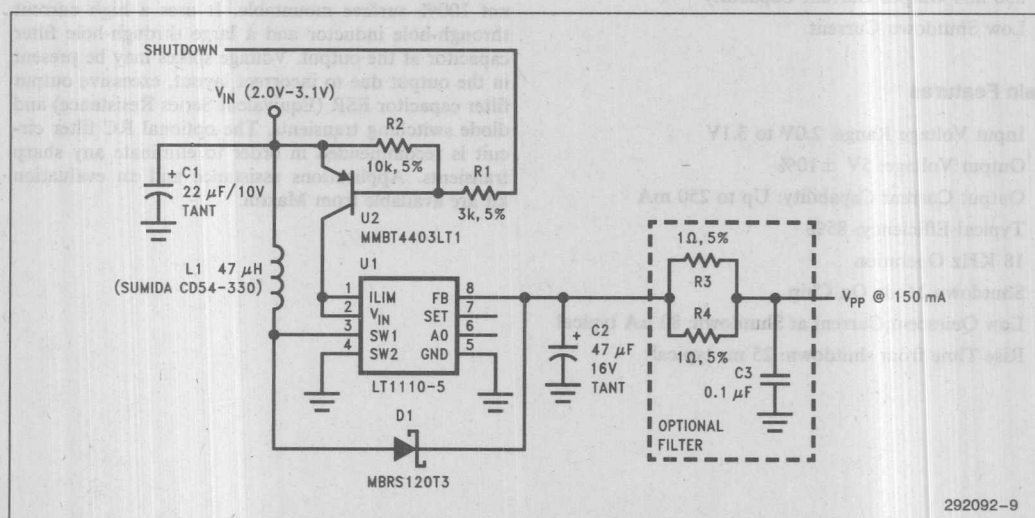


Figure 5-2. Linear Technology LT1110-5 3V to 5V Converter (150 mA)

Optimal Attributes

- Smallest Size
- Low Shutdown Current
- All Surface Mount

- Typical Efficiency: 76%
- 60 KHz Operation
- Shutdown Mode Using External Components
- Low Quiescent Current at Shutdown: 16 μ A typical
- Rise Time from Shutdown: 4 ms typical

Main Features

- Input Voltage Range: 2.0V to 3.1V
- Output Voltage: 5V \pm 5%
- Output Current Capability: Up to 150 mA

The LT1110-5 from Linear Technology is a fixed 5V version of the converter shown for the 12V design in Section 4.1.

Table 5-2. Parts List for the LT1110-5 3V to 5V Converter

Ref	Part #	Value/Type	Source	Cost*
U1	LT1110-5CS8	SMPS IC	Linear Tech (408) 954-8400	\$2.60
C1	267M1002- 226-MR-720	22 μ F/10V Tantalum Chip	Matsuo (714) 969-2491	\$0.23
C2	267M1602- 476-MR-720	47 μ F/16V Tantalum Chip	Matsuo (714) 969-2491	\$0.47
C3	GRM40Z5U104M050AD	0.1 μ F	Murata Erie (404) 436-1300	\$0.06
D1	MBRS120T3	Schottky Diode	Motorola (800) 521-6274	\$0.30
L1	CD75-330	33 μ H	Sumida (708) 956-0666	\$0.38
R1	9C08052A3001JLR	3 K Ω , 5%	Philips (817) 325-7871	\$0.02
R2	9C08052A1002JLR	10 K Ω , 5%	Philips (817) 325-7871	\$0.02
R3, R4	9C08052A1R00JLR	1 Ω , 5%	Philips (817) 325-7871	\$0.04
U2	MMBT4403LT1	PNP Transistor	Motorola (800) 521-6274	\$0.09
Total Cost				\$4.21

*Cost estimates based on published 10K unit pricing at the time this application note was written.

6.0 DOWN-CONVERTING TO 12V

The ability to down-convert to 12V from a higher voltage is often needed (as in the telecommunications environment). This section presents some good solutions for obtaining V_{pp} from a higher voltage.

6.1 Maxim Integrated Products MAX667

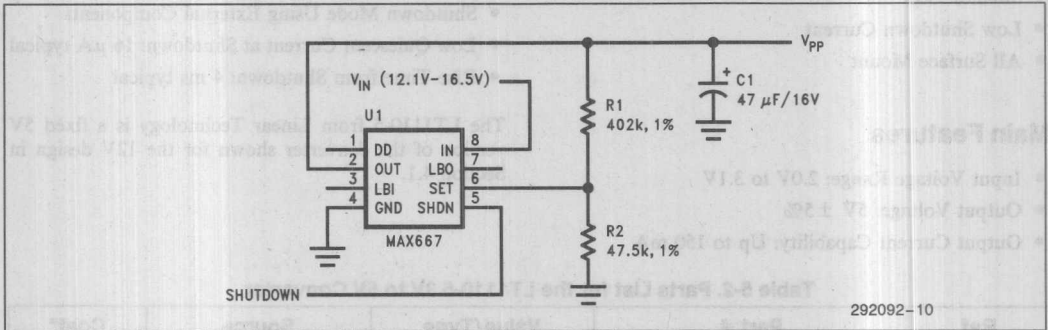


Figure 6-1. Maxim MAX667 12V Linear Voltage Regulator

Optimal Attributes

- Small Size
- Ultra Low Shutdown Current
- All Surface Mount
- Very Low Dropout

- Output Current Capability: Up to 120 mA
- Typical Efficiency: 70%
- Shutdown Mode On Chip
- Low Quiescent Current at Shutdown: 0.2 μ A Typical
- Rise Time from Shutdown: Less than 0.1 ms Typical

Main Features

- Input Voltage Range: 12.1V to 16.5V
- Output Voltage: 12V \pm 5%

Table 6-1. Parts List for the MAX667 12V Step Down Converter

Ref	Part #	Value/Type	Source	Cost*
U1	MAX667CSA	SMPS IC- SO8 Package	Maxim (408) 737-7600	\$2.10
C1	267M1602-476-MR-720	7 μ F/16V Tantalum	Matsuo (714) 969-2491	\$0.47
R1	9C08053A4023JLR	402 K Ω , 1%	Philips (817) 325-7871	\$0.03
R2	9C08053A4752JLR	47.5 K Ω , 1%	Philips	\$0.03
Total Cost				\$2.63

*Cost estimates based on published 10K unit pricing at the time this application note was written.

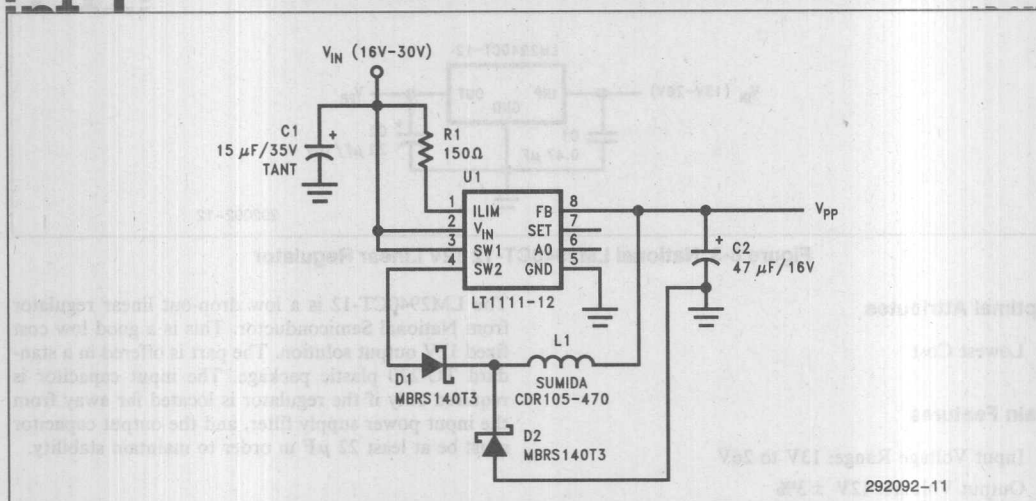


Figure 6-2. Linear Technology LT1111-12 Step Down Switcher

Optimal Attributes

- High Efficiency
- All Surface Mount

Main Features

- Input Voltage Range: 16V to 30V
- Output Voltage: 12V \pm 5%
- Output Current Capability: Up to 120 mA
- Typical Efficiency: 80%

Table 6-2. Parts List for the LT1111-12 12V Step Down Converter

Ref	Part #	Value/Type	Source	Cost*
U1	LT1111-12	SMPS IC- SO8 Package	Linear Tech (408) 432-1900	\$2.20
C1	267M3502-225-MR-720	2.2 μ F/35V Tantalum	Matsuo (714) 969-2491	\$0.28
C2	267M1602-476-MR-720	47 μ F/16V Tantalum	Matsuo (714) 969-2491	\$0.47
R1	9C08052A1500JLR	150 Ω , 5%	Philips (817) 325-7871	\$0.02
L1	CDR105-470	47 μ H	Sumida (708) 956-0666	\$0.38
D1, D2	MBRS140T3	Schottky Diode	Motorola (800) 521-6274	\$0.60
Total Cost				\$3.95

*Cost estimates based on published 10K unit pricing at the time this application note was written.

6.3 National Semiconductor LM2940CT-12

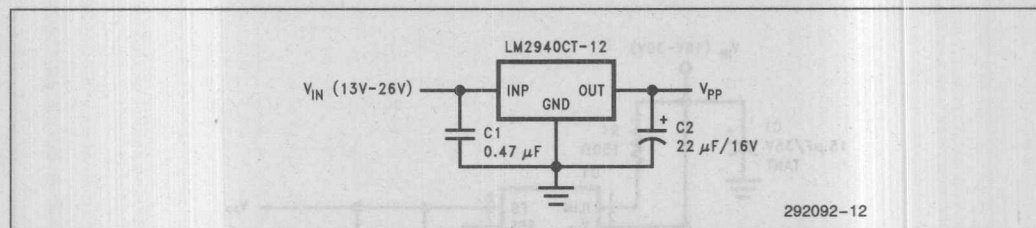


Figure 6-3. National LM2940CT-12 12V Linear Regulator

Optimal Attributes

- Lowest Cost

Main Features

- Input Voltage Range: 13V to 26V
- Output Voltage: 12V \pm 3%
- Output Current Capability: 1A

The LM2940CT-12 is a low drop-out linear regulator from National Semiconductor. This is a good low cost fixed 12V output solution. The part is offered in a standard TO-220 plastic package. The input capacitor is required only if the regulator is located far away from the input power supply filter, and the output capacitor must be at least 22 μ F in order to maintain stability.

Table 6-3. Parts List for the LM2940CT-12 Step Down Converter

Ref	Part #	Value/Type	Source	Cost*
U1	LM2940CT-12	Voltage Reg TO-220	National (408) 721-5000	\$0.95
C1	GRM43-2Z5U474M050AD	0.47 μ F/50V	Murata Erie (404) 436-1300	\$0.07
C2	267M1602-226-MR-720	22 μ F/16V Tantalum	Matsuo (714) 969-2491	\$0.28
Total Cost				\$1.30

*Cost estimates based on published 10K unit pricing at the time this application note was written.

7.0 OBTAINING V_{PP} FROM 12V UNREGULATED

In systems like the desktop computer, a 12V supply exists but may not be regulated to $\pm 5\%$. If this voltage is used as the V_{PP} source for flash memory, it may well degrade the write/erase performance of the memory, or adversely affect its reliability. Fortunately, in most of the situations where a 12V unregulated (or not regulated to within 5%) supply exists, a 5V supply also exists in the system (the desktop computer is a good example). It is recommended in such cases that the existing 5V supply be used to obtain the 12V $\pm 5\%$ rail. This approach is more economical, more efficient, and provides space savings over a buck-boost topology that takes unregulated 12V and regulates it to $\pm 5\%$.

In the rare case where a 5V supply is not present, modular solutions exist that will regulate the unregulated 12V supply to $\pm 5\%$. However, these are bulky and expensive. Moreover, many of them require that a minimum load be maintained in order to stay in regulation. One such solution is presented in Appendix A.

8.0 SUMMARY

For battery powered applications, the author views the discrete switching regulator IC solution as a better choice than the modular solution. The lower cost, higher efficiency, and smaller size/height associated with discrete solutions justify the small additional design effort required to incorporate them in flash memory applications. In applications where the primary source of power is a wall power outlet, or in applications where the flash memory will be written to infrequently, efficiency and quiescent current take on secondary importance. In such cases, it may be acceptable to use a 12V regulated (to within $\pm 5\%$) tap from the system supply. Alternatively, the ability to easily design-in modular solutions may outweigh the disadvantages of lower efficiency and increased cost. For those users wishing to incorporate modular solutions, Appendix A provides some of the lower cost solutions from this industry segment.

APPENDIX A MODULAR SOLUTIONS

Modular solutions may work well in non-battery powered situations where the efficiency of the power supply converter is not critical. These are also advantageous in that they usually do not need any external components and there is no converter design involved. However, the type and quality of the discrete components used in these hybrid solutions is open to question. This is not true in the case of the discrete converter designs presented in the earlier sections, where the quality of the components used are under the control of the system design engineer. Hence, even though modular solutions offer the convenience of a single package and ease of testability, the quality/reliability of comparably priced modular solutions may be questionable.

Some modular solutions suited to flash memory applications are presented below, with a brief description of each. Sources for obtaining these are listed in Appendix B.

A.1 International Power/Newport Components NMF0512S

The NMF0512S is a 5V to 12V hybrid power module that has an output current capability of 80 mA. Output tolerance is $\pm 5\%$. It is equipped with a shutdown pin which can be used to switch V_{pp} off. However, power dissipated in the shutdown mode is relatively high (about 100 mW). The part is small in size and measures 0.76 in. (19.5 mm) x 0.4 in. (9.8 mm) x 0.4 in. (9.8 mm), and costs about \$7.90 in 10K quantities (at the time this application note was written). Typical efficiency of conversion is 62%.

A.2 Xentek NPSC-0512S

The Xentek NPSC-0512S is a 1W power module that converts 5V to V_{pp} and will source up to 80 mA of continuous current. However, it uses two external filter capacitors—one at the input and one at the output. The input filter capacitor is 47 $\mu\text{F}/10\text{V}$, and the output filter capacitor is 100 $\mu\text{F}/16\text{V}$. Size of the solution (converter alone) is 0.87 in. (22 mm) x 0.39 in. (10 mm) x 0.79 in. (20 mm). The NPSC-0512S does not have a shutdown mode. The part costs around \$5.00 in 10K quantities (at the time this application note was written). Typical efficiency of conversion is 60%.

A.3 Shindengen America Inc. HDF-0512D

The HDF-0512D module from Shindengen will convert unregulated 12V to $12\text{V} \pm 5\%$. This part is a dual output part ($\pm 12\text{V}$), but only the +12V line is used. The conversion efficiency is high (75% typical), and the part will provide a regulated V_{pp} voltage from input voltages as low as 8V, and as high as 16.5V. A minimum load of 5 mA needs to be maintained to guarantee regulation. Size of the solution is 1.75 in. (44 mm) x 0.43 in. (11 mm) x 0.8 in. (20 mm). Cost is approximately \$10.00 in quantities of 10K (at the time this application note was written).

Ref #	Vendor Name	Part #	Input C (Volts)	Output V (Volts)	Output C (mA)	Efflc (%)	# Ext Comp (Note 1)	100% SMD ?	Cost (Note 2)	PC Area (Note 3)	Height (in)	SHDN ?	ISHDN (Note 4)	R Time (Note 5)	Temp
3.1	Maxim	MAX732	4V-7V	12V, 4%	120	90	5; D, L, 3C	Yes	\$3.93	0.56	0.18	Yes	70 μ A	1 ms	0°C, +70°C
3.2	Linear Tech	LT1110-12	5V, 10%	12V, 5%	120	76	7; D, L, T, 2R, 2C	Yes	\$4.58	0.45	0.20	Yes	16 μ A	1 ms	0°C, +70°C
3.3	Linear Tech	LT1109-12	5V, 10%	12V, 5%	60	84	4; D, L, 2C	Yes	\$3.61	0.38	0.18	Yes	375 μ A	1 ms	0°C, +70°C
3.4	Motorola	MC34063A	5V, 10%	12V, 5%	120	75	11; D, L, T, 3C, 5R	Yes	\$2.25	0.49	0.18	Yes	25 μ A	2 ms	0°C, +70°C
4.1	Linear Tech	LT1110-12	2V-3.1V	12V, 5%	30	70	7; D, L, T, 2R, 2C	Yes	\$4.71	0.45	0.18	Yes	16 μ A	4 ms	0°C, +70°C
4.2	Maxim	MAX732	1.8V-4V	12V, 4%	30	87	9; D, L, 7C,	Yes	\$4.80	0.7	0.18	Yes	45 μ A	25 ms	0°C, +70°C
4.3	Maxim	MAX732	1.8V-4V	12V, 4%	60	85	8; D, L, 6C,	No	\$4.15	1.11	0.49	Yes	45 μ A	75 ms	0°C, +70°C
5.1	Maxim	MAX658	2V-3.1V	5V, 5%	250	85	7; D, 2L, T, 3C	No	\$4.47	0.92	0.39	Yes	80 μ A	25 ms	0°C, +70°C
5.2	Linear Tech	LT1110-5	2V-3.1V	5V, 5%	150	76	7; D, L, T, 2R, 2C	Yes	\$4.72	0.45	0.20	Yes	16 μ A	1 ms	0°C, +70°C
6.1	Maxim	MAX667	12.1V-16V	12V, 5%	250	75	3; 2R, C	Yes	\$2.73	0.25	0.15	Yes	0.2 μ A	0.1 ms	0°C, +70°C
6.2	Linear Tech	LT1111-12	16V-30V	12V, 5%	120	80	6; 2D, L, 2C, R	Yes	\$3.95	0.78	0.2	No	N/A	N/A	0°C, +70°C
6.3	National	LM2940CT-12	13V-26V	12V, 3%	1A	12/V _{IN}	2; 2C	No	\$1.30	0.5	0.18	No	N/A	N/A	0°C, +70°C
A.1	International Power	NMF0512S	5V, 10%	12V, 5%	80	62	0	No	\$7.90	0.3	0.40	Yes	20 mA	10 μ s	-40°C, +70°C
A.2	Shindengen	HDF1212D	8V-16.5V	12V, 5%	120	77	0	No	\$10.00	0.76	0.80	No	N/A	N/A	-10°C, +70°C
A.3	Xentek	NPSC-0512S	5V, 10%	12V, 5%	80	60	2; 2C	No	\$5.50	0.34	0.79	No	N/A	N/A	-10°C, +70°C

NOTES:

- # External components. D: Diode, L: Inductor, C: Capacitor, R: Resistor, T: Transistor.
- Cost. Cost estimates assume 10K quantities, based on published pricing at the time this application note was written.
- PC Area. PC Aarea is conservatively estimated as 2.0x (area of all components). Where actual layouts are presented, the lower value is given. Note that this estimate is for a single sided board, and area can be reduced considerably if both sides of the board are utilized.
- I Shdn. Current consumed by supply at shutdown. Output settles to V_{CC} at shutdown, so some additional flash V_{pp} leakage/standby will exist.
- R Time. Rise time from shutdown state. Erase/Writes should not be attempted till V_{pp} level has risen to valid level after shutdown is disabled.

APPENDIX C

SOURCES/CONTACTS FOR RECOMMENDED DC-DC CONVERTERS

Linear Technology Corporation

Recommended Products:

- LT1110-12 (DC-DC Converter IC)
- LT1110-5 (DC-DC Converter IC)
- LT1109-12 (DC-DC Converter IC)
- LT1111-12 (DC-DC Converter IC)

In U.S.A.:

1630 McCarthy Blvd.
Milpitas, CA 95035-7487
Tel: (408) 432-1900
Fax: (408) 432-0507

In Europe (U.K.):

111 Windmill Road
Sunbury
Middlesex TW16 7EF
U.K.
Tel (44)(932) 765688
Fax (44)(932) 781936

In Asia (Japan):

4F Ichihashi Bldg
1-8-4 Kudankita Chiyoda-ku
Tokyo 102 Japan
Tel (81) (03) 3237-7891
Fax (81) (03) 3237-8010

Maxim Integrated Products

Recommended Products:

- MAX732 (DC-DC Converter IC)
- MAX658 (DC-DC Converter IC)
- MAX667 (DC-DC Converter IC)

In U.S.A.:

120 San Gabriel Drive
Sunnyvale, CA 94086
Tel (408) 737-7600
Fax (408) 737-7194

In Europe (U.K.):

Maxim Integrated Products (UK), Ltd.
Tel: (44) (734) 845255

In Asia (Japan):

Maxim Japan Co., Ltd.
Tel: 81 (03) 3232-6141

Motorola Semiconductor Inc.

Recommended Product:

- MC34063AD (DC-DC Converter IC)

In U.S.A.:

616 West 24th Street
Tempe, AZ 85282
Tel: (800) 521-6274

In Europe (U.K.):

Tel: (44) (296) 395-252

In Asia (Japan):

Tel: (81) (3) 440-3311

National Semiconductor

Recommended Product:

- LM2940CT-12 (Voltage Regulator IC)

In the U.S.:

2900 Semiconductor Drive
P.O. Box 58090
Santa Clara, CA 95052
Tel: (408) 721-5000

In Europe:

National Semiconductor (UK) Ltd.
The Maple, Kembrey Park
Swindon, Wiltshire SN26UT
U.K.
Tel: (07-93) 614141
Fax: (07-93) 697522

In Asia:

National Semiconductor Japan Ltd.
Sanseido Bldg. 5F
4-15 Nishi Shinjuku
Shinjuku-ku
Tokyo 160 Japan
Tel: (81) (3) 299-7001
Fax: (81) (3) 299-7000

Newport Components/ International Power

Recommended Product:

— NMF0512S (5V-12V Converter Module)

In U.S.A.:

International Power Sources
200 Butterfield Drive
Ashland, MA 01721
Tel: (508) 881-7434
Fax: (508) 879-8669

In Europe:

Newport Components
4 Tanners Drive
Blakelands North
Milton Keynes MK14 5NA
Tel: (0908) 615232
Fax: (0908) 617545

Shindengen Electric Co. Ltd.

Recommended Product:

— HDF0512D (12V unreg. to 12V reg. converter module)

In the U.S.:

2649 Townsgate Road #200
Westlake Village, CA 91361
Tel: (800) 634-3654
Fax: (805) 373-3710

In Europe:

Shindengen Magnaquest U.K. Ltd.
Unit 13, River Road,
Barking Business Park,
33 River Road, Barking,
Essex IG11 ODA
Tel: (44) (81) 591-8703
Fax: (44) (81) 591-8792

In Asia:

2-1,2-Chome Ohtemachi
Chiyoda-ku
Tokyo 100
Japan
Tel: (81) (3) 279-4431
Fax: (81) (3) 279-6478

Kentek Inc.

Recommended Product:

— NPSC0512S (5V-12V Converter Module)

In U.S.A.:

760 Shadowridge Drive
Vista, CA 92083
Tel: (619) 727-0940
Fax: (619) 727-8926

In Europe (Germany):

Kentek, Inc.
c/o Taiyo Yuden GMBH.
Obermaierstrasse 10,
D-8500 Nurnberg 10
Federal Republic of Germany
Tel: (49) (911) 350-8400
Fax: (49) (911) 350-8460

In Asia (Japan):

Kentek, Inc.
c/o Taiyo Yuden., Ltd.
6-16-20, Ueno, Taito-ku
Tokyo 110
Japan
Tel: (81) (3) 3837-6547
Fax: (81) (3) 3835-4752

APPENDIX D

CONTACTS FOR DISCRETE COMPONENTS

Matsuo Electric Co., Ltd.

Matsuo's 267 series surface mount tantalum chip capacitors are recommended by Maxim and Linear Technology for input and output filter capacitors on their DC-DC converters. Part #s are included on the parts list that accompanies most solutions. If alternate "equivalents" are required, choose high reliability, low ESR (Equivalent Series Resistance) and low ESL (Equivalent Series Inductance) type tantalums, which help in keeping output ripple and switching noise to a minimum.

In U.S.A.:

2134 Main St., Ste. 200
Huntington Beach, CA 92648
Tel: (714) 969-2491
Fax: (714) 960-6492

In Europe:

Steucon - Center II Mergenthalleralle 77
D-6236 Eschben/Ts.
Federal Republic of Germany
Tel: 6196-470-361
Fax: 6196-470-360

In Asia:

Oak Esaka Bldg.
10-28 Hiroshiba-Cho
Suita-shi
Osaka 564
Tel: (06) 337-6450
Fax: (06) 337-6456

Sumida Electric Co. Ltd.

Sumida CD series surface mount inductors are recommended by Maxim, Linear Technology for their miniature size and relatively low cost. These are well suited to low power DC-DC converter applications. Contact Sumida Electric directly for procuring these. The part #s are included in the parts list that accompanies most solutions. In applications where noise (EMI) is a concern, shielded varieties are also offered by Sumida.

In U.S.A.:

637 East Golf Road
Suite 209
Arlington Heights, IL 60005
Tel: (708) 956-0666
Fax: (708) 956-0702

In Asia:

4-8 Kanamachi 2-chome,
Katsushika-ku,
Tokyo 125
Japan
Tel: (81) (03) 3607-5111
Fax: (81) (03) 3607-5428

Coiltronix Inc.

Coiltronix is recommended as a good alternate source for surface mount inductors. The CTX series offered by Coiltronix is well suited to DC-DC converter applications. These are shielded, and have a toroidal core. However, they are bigger in size and currently much more expensive (7X to 8X) than the Sumida varieties recommended in the solutions herein. The equivalent part numbers are:

Sumida CD54-470 → Coiltronix CTX50-1
Sumida CD54-180 → Coiltronix CTX20-1
Sumida CD54-220 → Coiltronix CTX20-1
Sumida CD75-470 → Coiltronix CTX50-2
Sumida CDR105-470 → Coiltronix CTX50-2

In U.S.A.:

Coiltronix Inc.
984 S.W. 13th Court
Pompano Beach, FL 33069
Tel: (305) 781-8900
Fax: (305) 782-4163

In U.K.:

Microelectronics Technology Ltd.
Great Haseley Trading Estate
Great Haseley
Oxfordshire OX9 7PF
U.K.
Tel: (08) 44 278781
Fax: (08) 44 278746

In Asia:

Serial System Mktg.
Poh Leng Bldg., #02-01
21 Moonstone Lane
Singapore 1232
Tel: 2938830
Fax: 2912673

Coilcraft

Coilcraft is also recommended as a good alternate source for surface mount inductors. The N2724-A shielded series is well suited to DC-DC converter applications. These are bigger and currently more expensive (2x to 3x) than the Sumida inductors recommended in the solutions. Contact Coilcraft directly for any applications assistance or for procurement of these parts. The equivalent part numbers are:

Sumida CD54-470 → Coilcraft N2724-A 47 μ H
Sumida CD54-180 → Coilcraft N2724-A 18 μ H
Sumida CDR105-470 → Coilcraft N2724-A 47 μ H

In the US:

1102 Silver Lake Road
Cary, IL 60013
Tel: (708) 639-6400
Fax: (708) 639-1469

In Europe:

21 Napier Place
Wardpark North
Cumbernauld
Scotland G68 0LL
Tel: 0236 730595
Fax: 0236 730627

In Asia:

Block 101, Boon Keng Road
#06-13/20
Kallang Basin Industrial Estate
Singapore 1233
Tel: 2966933
Fax: 2964463

Philips Components

Philips Components is recommended as a good source for surface mount (SMD) resistors (standard 9C series, and 9B (MELF) series). Part #s are included in the parts list that accompanies most of the solutions in the application note. Many alternate sources exist.

In the US:

2001 W. Blue Heron Blvd.
P.O. Box 10330
Riviera Beach, FL 33404
Tel: (407) 881-3200
Fax: (407) 881-3304

In Europe:

Philips Components Ltd.
Mullard House
Torrington Place
London WC1E 7HD
Tel: (44) 71 580 6633
Fax: (44) 71 636 0394

In Asia:

Philips K.K.
Philips Bldg. 13-37
Kohnan 2-chome
Minato-Ku Tokyo 108
Tel: (81) 3 740-5028
Fax: (81) 3 740-5035

Siliconix-Logic Level PFETs

Siliconix offers low-"on" resistance logic level PFETs (Si9400, and Si9405) that can be used for switching a DC-DC converter into a shutdown state by using these switches on the high side of the input to the converter (see Appendix E).

In the US:

2201 Laurelwood Road
P.O. Box 54951
Santa Clara, CA 95056-9951
Tel: (408) 988-8000
Fax: (408) 727-5414

In Europe:

Weir House
Overbridge Square, Hambridge Lane
Newbury, Berks RG14 5UX
Tel: (0635) 30905
Fax: (0635) 34805

In Asia:

Room 709, Chinachem Golden Plaza
77 Mody Road
TST East Kowloon
Tel: (852) 724-3377
Fax: (852) 311-7909

APPENDIX E OTHER DESIGN CONSIDERATIONS

E.1 V_{pp} Valid Handshake Logic

It is often desirable to have, along with the V_{pp} solution, a handshake signal (using extra hardware) that is asserted as long as the voltage level on V_{pp} is valid. The following schematic illustrates a good way of achieving this. This handshake signal could be used to determine when it is suitable to perform writes/erases on the flash device. The circuit shown uses a precision zener voltage reference and a comparator, along with bias resistors, to monitor the voltage level on V_{pp}. The point at which the comparator trips must be set after careful consideration of the variation in the reference voltage and the tolerances on the bias resistors. The worst case conditions on these variations must guarantee that the handshake signal is asserted when V_{pp} is at its worst case lower-end level (11.4V). Care must be taken to use the exact same components as specified in order to maintain the tight tolerance on the trip level of the output signal.

E.2 Obtaining Shutdown Using Logic Level PFETs

Low “on” resistance logic level PFETs can be used on the high side of the input to the DC-DC converters to obtain shutdown. One such part is the Si9405 from Siliconix Inc. The device is part of the “little foot” series, and is available in an SO8 (8-pin surface mount) package. The Si9405 is a logic level PFET with an “on re-

sistance” of 0.2Ω (at a gate drive of 4.5V). It is important to have as low an “on” resistance as possible, since the peak currents and start-up currents into the supply are high. Care must be taken to ensure that the DC-DC conversion process is not affected after accounting for the drop in input voltage across the PFET.

E.3 Working of the Discrete Step Up Switching Regulator

This section presents a brief overview of the operation of discrete step up switching regulators, and presents issues that the user needs to be concerned with while designing these solutions into the system.

The four most basic elements of a discrete switching regulator power supply are:

1. The SMPS IC (which includes the switch control element and logic, along with the power switch itself),
2. An inductor for storage and transfer of energy between the input and output,
3. A switching diode to direct the inductor energy to “catch”, or channel, the inductor energy to the output, and
4. An output filter capacitor.

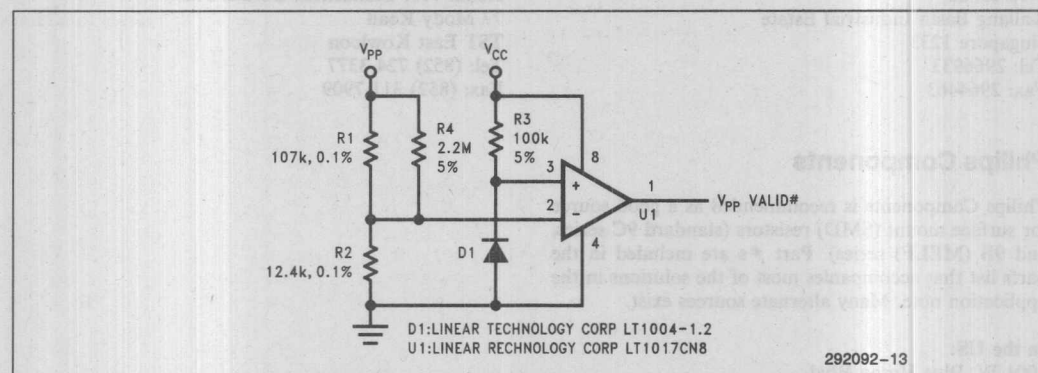


Figure E-1. V_{pp} Valid Handshake Circuit

In the boost configuration where the output voltage is greater than the input voltage, the basic switching power supply configuration is as shown in Figure E.2:

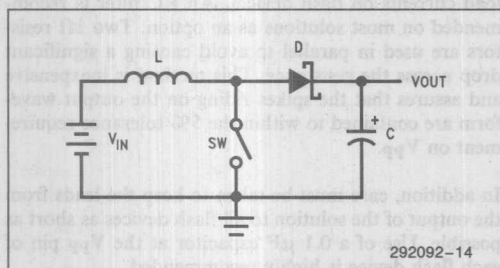


Figure E-2. Working of the Step-Up Switching Regulator

The power switch SW can be turned on and off; the control for it is derived from a feedback mechanism that senses the output voltage. While the switch is turned on, the inductor stores energy as the current flows through it from the input supply. The peak current through the inductor I_L can be approximated as $(V_{IN}/L * t_{ON})$; where t_{ON} is the on time of the switch. During this time, the energy is supplied by the input voltage, $V_L = V_{IN}$. The output is isolated from the inductor via the reverse-biased diode, and the load current is supplied by the output filter capacitor. When the switch turns off, the energy stored in the inductor appears as a rapidly increasing voltage across the inductor. As soon as this voltage reaches a value equal to the output voltage plus the voltage drop across the diode, the diode switches on and current starts to flow through the diode. This diode current supplies the load current while also at the same time charging up the output filter capacitor to the output voltage.

The switch is controlled by sensing the output voltage via a feedback mechanism—usually a pair of resistors. This sense voltage is gated via a comparator whose output acts as a control signal to an oscillator. The oscillator output controls the switch.

The power into the inductor P_L can be approximated as:

$$P_L = 0.5 * L * I_{PK}^2 * f_{OSC}$$

and the power into the load P_{LOAD} (out of the inductor) can be approximated as

$$P_{LOAD} = (V_{OUT} + V_D - V_{IN}) * I_{OUT}$$

The peak currents through the inductor is usually several times higher than the load current, is mostly of the value of the load current and builds up during time t_{ON} . On most of the solutions presented here, peak operating currents lie in the range of 500 mA to 1.2A. Though this may seem high, most of this in-rush of energy is transferred to the output, and little is lost to heat due to the efficient energy storage characteristic of inductors. Note that since the peak currents are high, the input voltage source must be capable of providing this current, and the current capability of the input source must not be calculated simply as $(V_{OUT} * I_{OUT}) / (V_{IN} * \text{Eff})$. A large bypass capacitor at the input pin of the converter is hence also necessary on all designs.

Some of the solutions presented in this application note are of the fixed duty cycle or fixed on time type (e.g. LT1110-12, LT1109-12, MC34063A), whereas some of them vary the duty cycle depending on the load current (e.g. MAX732, MAX658). These latter ones provide higher efficiencies.

2

Inductor Selection

The choice of an inductor is crucial to the design of the power supply system. To begin with, the inductor value must be low enough to supply the peak currents needed when the input voltage V_{IN} , as well as the on time t_{ON} , are at their worst case low value. On the other hand, the inductor value must be high enough so that the peak currents at the worst case high values do not exceed the maximum peak currents that can be handled by the switch. Furthermore, once the value has been picked, the physical inductor that is chosen for the job must be able to handle these peak currents, and must not saturate. This is done by picking an inductor whose DC current rating is more than the worst case peak current that will be required by the operation of the device. The other characteristic to consider is the resistance of the inductor. In order to keep losses to a minimum, it is essential that the resistance of the coil is a minimum. Thus, it is important to use the inductors specified in the parts list that accompanies the solutions. These have been carefully chosen after reviewing the requirements. Alternate inductors may be used, as long as they are "equivalent".

EMI Concerns

Since the switching regulators presented in this application note switch at frequencies between 60 KHz and 200 KHz, there exists a potential for EMI. In cases where EMI may be a problem, shielded inductors can be used. This will reduce EMI significantly. Shielded versions of the inductors specified are readily available. Contact the vendor directly for these.

Output Switching Noise

Output switching noise has several sources. The most significant one is the IR drop through the ESR (Equivalent Series Resistance) of the output filter capacitor. This is caused by switching current pulses from the inductor. There is also noise in the form of switching spikes riding on the DC output. This is due to the output filter capacitor's ESL (Equivalent Series Inductance), current spikes in the ground trace and rectifier turn-on transients.

It is important to use low ESR and low ESL output and input filter capacitors. Proper layout is also essential in

order to avoid spikes in the output. The safest solution is to use a filter circuit at the output. LC filters are not recommended, because of the transient nature of the load currents on flash devices. An RC filter is recommended on most solutions as an option. Two 1Ω resistors are used in parallel to avoid causing a significant drop across the resistance. This method is inexpensive and assures that the spikes riding on the output waveform are contained to within the 5% tolerance requirement on V_{DD} .

In addition, care must be taken to keep the leads from the output of the solution to all flash devices as short as possible. Use of a 0.1 μF capacitor at the V_{PP} pin of each flash device is highly recommended.

APPENDIX F PC LAYOUTS FOR SOME RECOMMENDED SOLUTIONS

F.1 Maxim Integrated Products MAX732

The double-sided layout presented below (Figure F-1) has been designed for the MAX732 5V–12V converter solution (Section 3.1). It is a double sided layout and has been designed for the parts specified in the parts list that accompanies the solution. Contact Maxim for any additional layout assistance.

F.2 Linear Technology Corporation LT1110-12

The single-sided layout presented below (Figure F-2) can be used to implement the LT1110-12 5V to 12V

converter (Section 3.2), the LT1110-12 3V–12V converter (Section 4.1), or the LT1110-5 3V to 5V converter (Section 5.2). The layout has been designed for the parts that are specified in the parts list that accompanies these solutions. Contact Linear Technology for any additional layout assistance.

F.3 Linear Technology Corporation LT1109-12

The single-sided layout presented below (Figure F-3) can be used to implement the LT1109-12 5V–12V converter solution (Section 3.3). The layout has been designed for the parts that are specified in the parts list that accompanies the solution. Contact Linear Technology for any additional layout assistance.

2

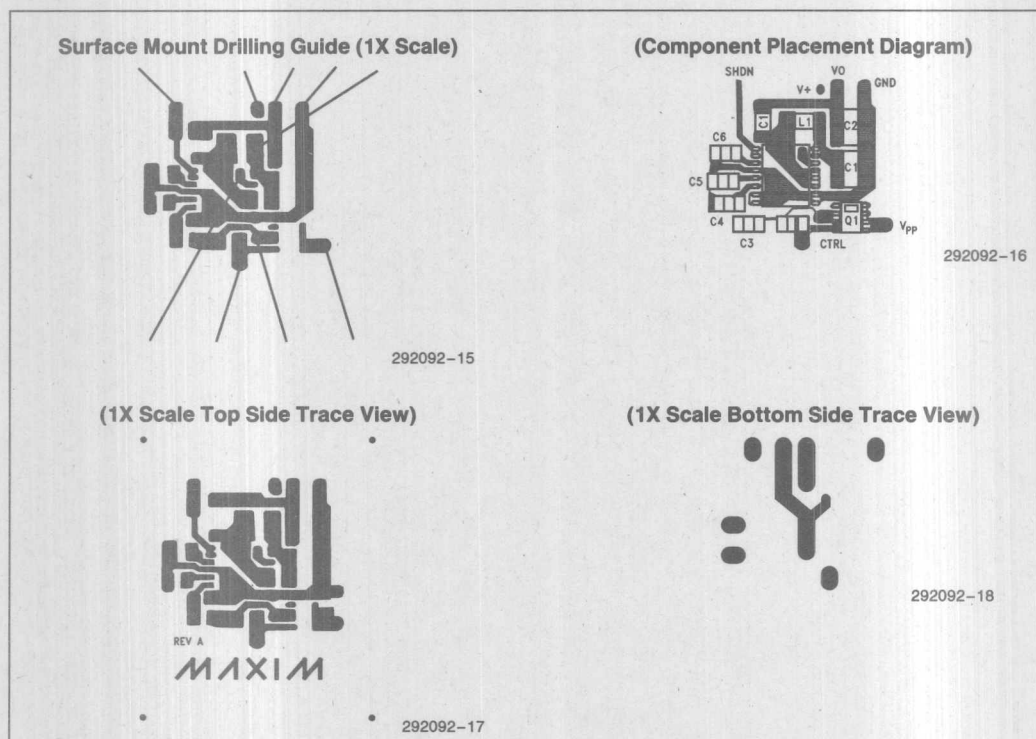


Figure F-1

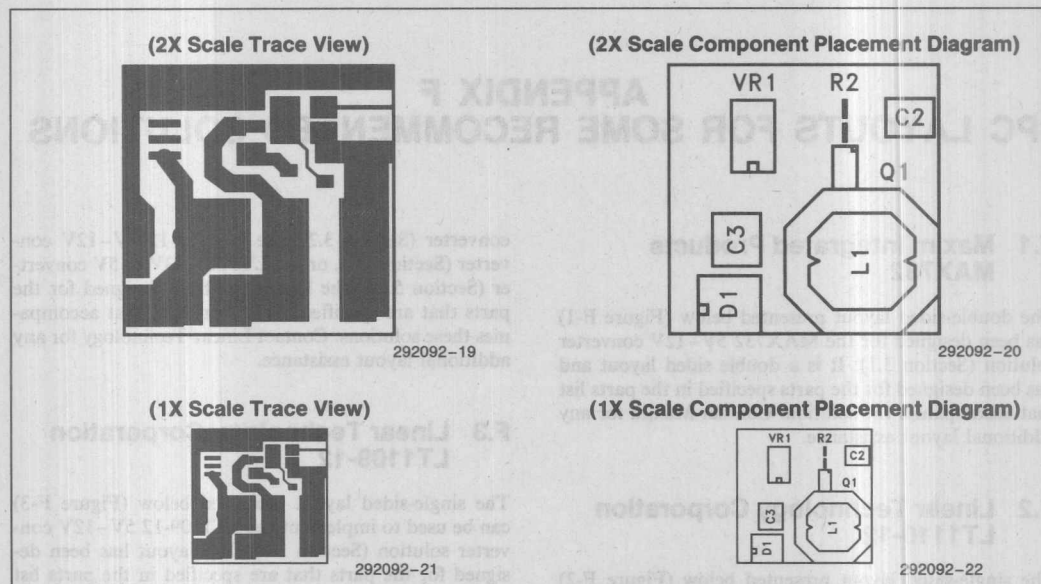


Figure F-2

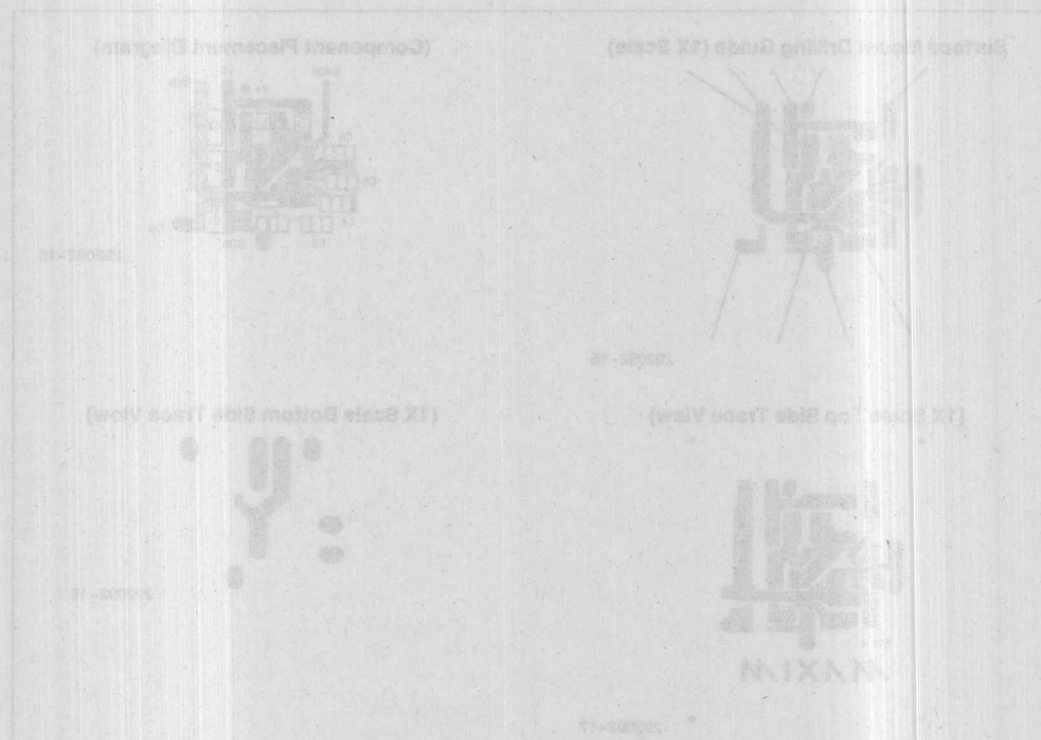


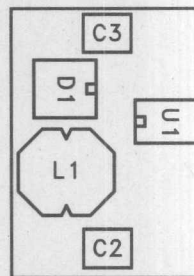
Figure F-1

(2X Scale Trace View)



292092-25

(2X Scale Component Placement Diagram)



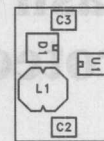
292092-26

(1X Scale Trace View)



292092-27

(1X Scale Component Placement Diagram)



292092-28

Figure F-3

NOTE

Flash Memory Write Protection Techniques

BRIAN DIPERT
SENIOR TECHNICAL MARKETING ENGINEER

September 1993

Flash Memory Write Protection Techniques

CONTENTS	PAGE
1.0 INTRODUCTION	2-44
2.0 WHY IS WRITE PROTECTION IMPORTANT?	2-44
3.0 SYSTEM WRITES WITH BULK-ERASE FLASH MEMORIES	2-44
4.0 SYSTEM WRITES WITH BOOTBLOCK AND FlashFile™ MEMORIES	2-45

CONTENTS	PAGE
5.0 PREVENTING UNINTENDED WRITES DURING NORMAL SYSTEM OPERATION	2-45
6.0 PREVENTING UNINTENDED WRITES DURING SYSTEM POWERUP/POWERDOWN AND RESET	2-46
6.1 Designing for Flash Memory System Power Sequencing Protection	2-47
7.0 SUMMARY	2-48

2

3.0. SYSTEM WRITES WITH BULK-ERASE FLASH MEMORIES

First-generation bulk erase flash memories from Intel Corporation are shown in Figure 1. These devices naturally power up in a "Flash Array" mode in which they output array data when read transitions to the flash array mode. When a write command is issued, the data is written to the array. This mode is not suitable for system writes because the data is written to the array and not to the memory.

Device	Density
28F258A	32 Kbytes (x8)
28F2512	64 Kbytes (x8)
28F010	128 Kbytes (x8)
28F020	256 Kbytes (x8)

Figure 1. Intel Corporation Bulk-Erase Flash Memories

Bulk-erase flash memories include several forms of "protection" to guard against unintended writes. Writes with V_{pp} (the programming voltage) at V_{pp} (0V to 5.7V) are distinguished by the flash memory. Similarly, write attempts with V_{pp} at or below V_{pp} (0.3V on most devices) are ignored. Finally, there are three regular multi-byte command sequences to initiate internal program or erase algorithms. Note, however, that while the erase command sequence (shown in Figure 2) requires both the proper Flash Setup and Erase (E) register bits, the program sequence (Figure 3) requires only the valid program setup command. The second command in the latter sequence can have any value and is interpreted as data to be programmed. This means that if the flash memory receives an invalid program setup command, the very next write to the device (provided it will be executed as programmed data and initiates an internal program event (V_{pp} is above V_{pp})).

3.0. WHY IS WRITE PROTECTION IMPORTANT?

Let's begin by identifying the key characteristics of two generic memory technologies: ROM (Read-Only Memory) and RAM (Random-Access Memory). Flash memory contains many of the capabilities of both in one solution. Therefore, it is often being utilized to replace ROM and/or RAM in new designs. At a minimum, flash memory's status as a relatively new technology means that many engineers are moving to it from the familiarity of a ROM/RAM knowledge base.

RAM is fully erasable on a bit-by-bit basis, and the mechanism for writing to it is established and well understood. RAM is in-system updatable, yet it is volatile. This means that when a RAM memory loses power, it also loses its data. RAM is guaranteed not to contain valid information on powerup.

ROM offers the advantage of nonvolatility, i.e. when power is removed from the device, the information stored remains unchanged. However, ROM is not re-programmable. Once the information is initially put into the device, it is permanent and nonerasable. To replace the information, you have to physically remove/reprogram the device itself.

Traditional system memory architectures often include both ROM (nonvolatile but non-updatable) and RAM (volatile but in-system updatable). The new model for system design retains some RAM for temporary data storage, but replaces the rest of RAM and ROM with flash memory. Being both nonvolatile and in-system updatable, flash memory encompasses the

1.0 INTRODUCTION

Flash memory's combination of nonvolatility and easy in-system updateability are key attributes driving its adoption into today's system designs. However, this flexibility also brings with it the responsibility (for hardware and software engineers) to ensure that writes to flash memory occur *only when intended*. This is especially important for those who are accustomed to designing with various ROM (nonvolatile but non-updateable) and RAM (updateable but volatile) memories.

This application note discusses techniques for proactively designing systems to prevent unintentional writes to flash memory. These design techniques are by no means complex or costly, but their implementation is crucial to ensuring reliable operation through system lifetime. For more information on the devices and specifications discussed in this document, please consult specific flash memory datasheets.

2.0 WHY IS WRITE PROTECTION IMPORTANT?

Let's begin by identifying the key characteristics of two generic memory technologies: ROM (Read-Only-Memory) and RAM (Random-Access-Memory). Flash memory combines many of the capabilities of both in one solution. Therefore, it is often being utilized to replace ROM and/or RAM in new designs. At a minimum, flash memory's status as a relatively new technology means that many engineers are moving to it from the familiarity of a ROM/RAM knowledge base.

RAM is fully alterable on a bit-by-bit basis, and the mechanism for writing to it is established and well understood. RAM is in-system updateable, yet it is volatile. This means that when a RAM memory loses power, it also loses its data. RAM is guaranteed *not* to contain valid information on powerup.

ROM offers the advantage of nonvolatility, i.e. when power is removed from the device, the information stored inside is retained. However, ROM is not in-system updateable. Once the information is initially put into the device, it is permanent and unchangeable. To replace the information, you have to physically remove/replace the device itself.

Traditional system memory architectures often included both ROM (nonvolatile but non-updateable) and RAM (volatile but in-system updateable). The new model for system design retains some RAM for temporary data storage, but replaces the rest of RAM and ROM with flash memory. Being both nonvolatile and in-system updateable, flash memory encompasses the

strengths of both RAM and ROM, offering new system architecture possibilities. However, whereas in the past RAM was guaranteed to be invalid on system powerup and ROM was guaranteed to be unalterable, the same cannot be said for flash memory.

Any alteration of flash memory contents (whether planned or unintended) is permanent regardless of system power transitions, until the data is again modified. As we'll see later, command writes to flash memory can also put it in modes where it outputs something other than array data, a non-permanent but still undesirable condition when not intended. This means that the system hardware and software must ensure that flash memory is written only when specifically desired, to ensure a predictable system environment. The following sections will discuss how this can be accomplished.

3.0 SYSTEM WRITES WITH BULK-ERASE FLASH MEMORIES

First-generation bulk erase flash memories from Intel Corporation are shown in Figure 1. These devices automatically power up in a "Read Array" mode in which they output array data when read. Transitions to alternate modes occur by writing commands to the flash memory.

Device	Density
28F256A	32 Kbytes (x8)
28F512	64 Kbytes (x8)
28F010	128 Kbytes (x8)
28F020	256 Kbytes (x8)

Figure 1. Intel Corporation
Bulk-Erase Flash Memories

Bulk-erase flash memories include several forms of "protection" to guard against unintended writes. Writes with V_{pp} (the program/erase voltage) at V_{PPL} (0V to 6.5V) are disregarded by the flash memory. Similarly, write attempts with V_{CC} at or below V_{LKO} (2.5V on most devices) are ignored. Finally, these devices require multi-byte command sequences to initiate internal program or erase algorithms. Note, however, that while the erase command sequence (shown in Figure 2) requires both the proper Erase Setup and Erase Confirm commands, the program sequence (Figure 3) relies only on the valid Program Setup command. The second command in the latter sequence can have any value, and is interpreted as data to be programmed. This means that if the flash memory receives an unintended Program Setup command, the very next write to the device (intended or not) will be interpreted as program data and initiate an internal program event (if V_{pp} is above V_{PPL}).

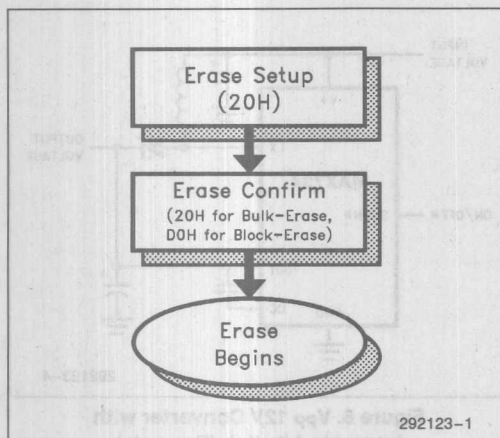


Figure 2. Flash Memory Erase Command Sequence (Simplified)

Beyond the program and erase sequences, the Read Intelligent Identifier Codes command will, when written to the flash memory, put it in a mode where it outputs device signature IDs instead of array information when read.

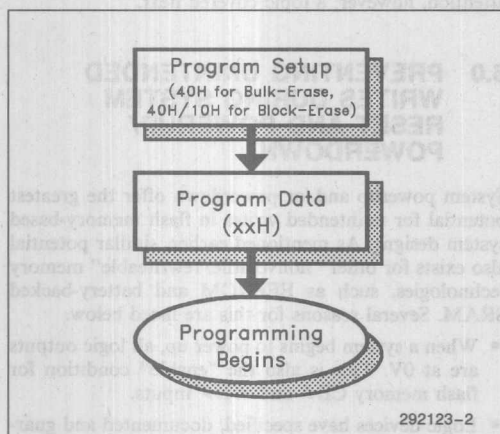


Figure 3. Flash Memory Programming Command Sequence (Simplified)

4.0 SYSTEM WRITES WITH BOOT BLOCK AND FlashFile™ MEMORIES

Second-generation block-erase Boot Block and FlashFile memories from Intel Corporation are shown in Figure 4. They function similarly to the bulk-erase devices described earlier, with a few key enhancements. As before, these devices automatically power up in "Read Array" mode, and transition to alternate modes via command writes.

BOOT BLOCK ARCHITECTURE	
Device	Density
28F001BX	128 Kbytes (x8)
28F200BX	256 Kbytes (x16)
28F002BX	256 Kbytes (x8)
28F400BX	512 Kbytes (x16)
28F004BX	512 Kbytes (x8)
FlashFile™ ARCHITECTURE	
Device	Density
28F008SA	1 Mbyte (x8)

Figure 4. Intel Corporation Block-Erase Flash Memories

For full access to the flash memory Status Register, as well as for enhanced interface to internal device identifiers, these block-erase flash memories will accept commands written to them regardless of V_{pp} voltage, as long as V_{CC} is above V_{LKO} . Program and erase algorithms initiated by command sequences will terminate with Status Register error indication and unaltered array data, if V_{pp} is at V_{pPL} . However, regardless of V_{pp} level, the device will still transition to a "Read Status Register" mode after program/erase command sequences are written. In this case, it will output data that the system, if the write was unintended, will not expect. The same multi-byte command sequences (shown in Figures 2 and 3) are used as in bulk-erase flash memories.

Boot Block and FlashFile memories provide commands (in addition to the program and erase sequences) which transition the memory to alternate modes, outputting data other than array information for subsequent reads. In this respect, they are similar to bulk-erase flash memory discussed earlier. These commands are Intelligent Identifier and Read Status Register.

Block-erase devices include a hardware input called $RP\#$ (or Reset/Powerdown). Among its many uses, this pin acts as a "master on/off switch" to completely disable the flash memory and lock all other control inputs. $RP\#$ is extremely effective at blocking unintended writes during system power transitions. This technique will be covered in detail, in a few paragraphs.

5.0 PREVENTING UNINTENDED WRITES DURING NORMAL SYSTEM OPERATION

Preventing unintended writes to flash memory during normal system operation is a routine part of debugging a new design, and a common concern for any "writable" device on the processor interface. Any combina-

tion of active chip select ($CE\#$) and active write enable ($WE\#$) has the potential of being decoded by the flash memory as a valid write attempt. One common culprit in these situations is the chip select decoder logic (PAL, etc.) between the processor and external devices. As addresses propagate through this logic at the beginning of an access cycle, or in the undefined address state between accesses, spurious chip selects of indeterminate duration can be generated. System hardware should ensure that at these times, $WE\#$ to flash memory stays at a logic "1" and doesn't transition low.

Some concern has also been expressed in the past about unintended writes in certain "open" systems such as the personal computer. In these environments, the type and function of software run on the machine is beyond the control of the computer manufacturer, who must accordingly design his/her hardware. For example, a third-party software utility may write to flash memory assuming DRAM at that location. More malicious, of course, is the case of the computer virus. Fortunately, in cases like this, hardware design to prevent unintended writes is fairly simple.

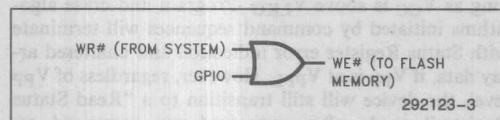


Figure 5. $WE\#$ Gating

Figure 5 shows one means of clarifying the $WE\#$ signal. When flash memory is used for BIOS storage, for example, the manufacturer's update utility is the only software that should be writing to the device. By toggling the general purpose I/O line (whose default state is, of course, "disabled"), the update utility can control whether writes from the system are blocked or allowed to pass to the flash memory. This type of $WE\#$ clarifying function is integrated in the Intel386TMSL and Intel486TMSL Microprocessor Supersets. ASICs integrating motherboard functions should also be designed to include such logic.

One other method for preventing flash memory alteration is by controlling (or "switching") the V_{pp} voltage, turning it on to V_{ppH} only when desired for system update. Many 12V converters and power supplies integrate this on/off function as shown in Figure 6, or it can be provided by an external FET. This approach will be used again in the next section on write protection during system power transitions. Note, however, that although it prevents actual flash memory data alteration, V_{pp} control is insufficient to keep block-erase flash memories from transitioning to alternate data output modes by unintended writes.

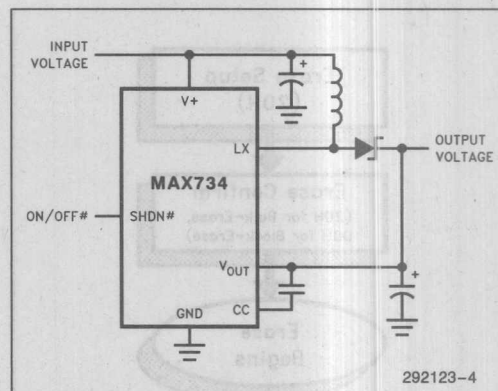


Figure 6. V_{pp} 12V Converter with Integrated Switch (Example)

In a traditional "closed" system, the software directing the hardware is totally under control of the system manufacturer. No additional effort should be needed (after the initial prototype hardware and software debugging) to protect the flash memory from unintended writes during normal system operation. Write control during system powerup and powerdown also requires attention, however; a topic covered next.

6.0 PREVENTING UNINTENDED WRITES DURING SYSTEM RESET AND POWERUP/POWERDOWN

System powerup and/or powerdown offer the greatest potential for unintended writes in flash memory-based system designs. As mentioned earlier, similar potential also exists for other "nonvolatile/rewriteable" memory technologies, such as EEPROM and battery-backed SRAM. Several reasons for this are listed below.

- When a system begins to power up, all logic outputs are at 0V. This is also the "enable" condition for flash memory $CE\#$ and $WE\#$ inputs.
- Logic devices have specified, documented and guaranteed operation only at a specific supply voltage range (typically $5V \pm 10\%$ or $3.3V \pm 0.3V$). Operation beyond this voltage range is not guaranteed and may not be consistent. Specifically, device output behaviour is typically undefined.
- Similarly, logic operation is sometimes undefined and erratic when devices are being reset. For example, MCS-186 embedded processors, when reset, tri-state their $WR\#$ (write enable) outputs, which will then typically drift toward 0V (or "enabled", to TTL inputs).

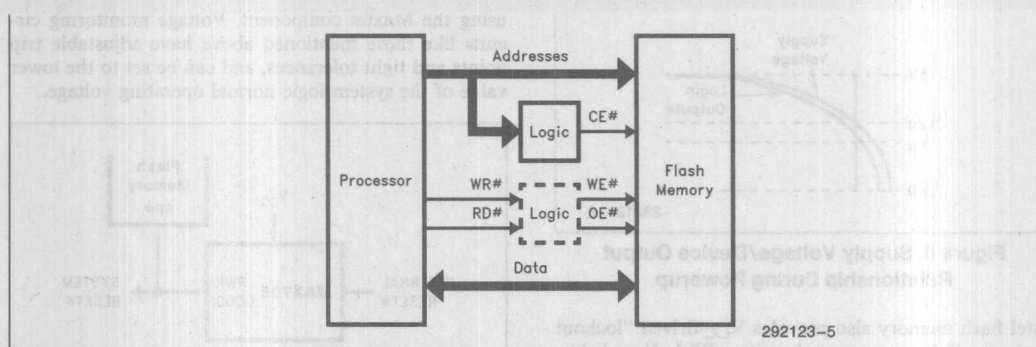


Figure 7. Basic Processor/Flash Memory Interface

- If both the V_{CC} and V_{pp} power supplies are switched “on” at the same time, one or the other is likely to ramp to a “valid” level first, depending on the relative capacitive loading at the supply outputs. Similarly, one supply will often ramp below its valid voltage range before the other, on system poweroff. This situation is acceptable, as long as the $WE\#$ and/or $CE\#$ signals to the flash memory are controlled.

Figure 7 shows a very basic example processor/memory interface. When the system power is switched on, the processor (or logic) $WE\#$ output and logic $CE\#$ output are both at GND. Depending on the processor and logic, these outputs may not reliably stabilize until V_{CC} ramps to 4.5V. In most cases, CPU and logic outputs will smoothly follow the supply voltage up to operating levels. Any oscillations on these outputs, however, can be decoded as a valid write by the flash memory, which begins to “wake up” below 4.5V V_{CC} . Similarly, address and data processor outputs are typically undefined below operating voltage ranges. Given a x8 interface between processor and flash memory (therefore, with 256 possible combinations of data inputs), there is a finite chance that a valid command byte will be randomly generated and written to the flash memory.

If the V_{pp} power supply output is less capacitively loaded than V_{CC} , V_{pp} can ramp above V_{ppL} before V_{CC} reaches 4.5V. This can cause unintended flash memory program and erase if the correct command data values are “spuriously” written to the device.

Again referencing Figure 7, the behaviour of processor/logic $CE\#$, $WE\#$ and address/data outputs are typically undefined once V_{CC} drops below 4.5V. If the power supply V_{pp} output is more capacitively loaded than V_{CC} , V_{pp} can remain above V_{ppL} as V_{CC} decays toward 0V. This has the potential to initiate program/erase operations in response to unintended flash memory writes.

6.1 Designing for Flash Memory System Power Sequencing Protection

Intel has taken several steps with respect to its flash memory designs to significantly minimize the possibility of an unwanted write during system powerup or powerdown. By synergizing system designs to these flash memory features, you can easily eliminate the potential for unwanted flash memory mode switching and/or data alteration.

Flash memories from Intel are guaranteed *not* to program or erase with V_{pp} below 6.5V. First generation bulk-erase devices additionally block *all* write attempts with V_{pp} below 6.5V. The implication here is clear; if possible, don’t switch on V_{pp} until after the system V_{CC} is stable (on powerup), and switch off V_{pp} before the system is powered down. The V_{pp} supply itself can be switched on/off, or an inline FET switch can be installed between the power supply output and flash memory input and controlled via an I/O line from the processor or discrete logic. Figure 6 gives an example of circuitry for the former case.



Figure 8. Supply Voltage/Device Output Relationship During Powerup

Intel flash memory also provides V_{CC} -driven "lockout protection" from unwanted writes. With V_{CC} below V_{LKO} , all write attempts to the flash memory are ignored. V_{LKO} varies between 2.5V and 2.0V depending on the specific flash memory, and its value is targeted to take advantage of the fact that in most cases device outputs closely follow V_{CC} inputs (both up and down). Referencing Figure 8, when V_{CC} exceeds V_{LKO} , device outputs will in most cases also be at approximately V_{LKO} , and consequently at a TTL "1" level (or disabled). The flash memory "protects itself" up to V_{LKO} , and the system designer must above that point ensure that flash memory control inputs are stable. Similarly, the flash memory is again protected once V_{CC} drops below V_{LKO} on system powerdown.

The RP# input (formerly known as PWD#), available on Intel Boot Block and FlashFile memories, acts as a "master on/off switch" for the device. With RP# at V_{IL} , the flash memory is put in a very low power mode called Deep Powerdown, and is essentially turned "off". In this state, all write attempts to the flash memory are disregarded. RP# can be driven by the POWERGOOD output of the system power supply (if this output exists) or from an external analog "power supply monitoring" device like the Maxim MAX705 or Motorola MC34064, providing absolute flash memory protection. Figure 9 gives an example system design

points and tight tolerances, and can be set to the lower value of the system logic normal operating voltage.

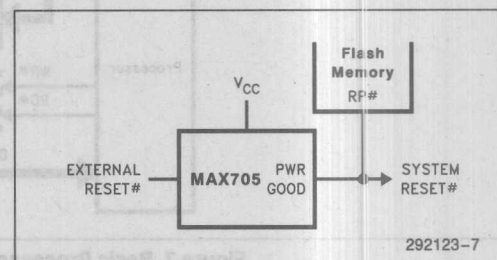


Figure 9. Reset Control during System Powerup and Powerdown

7.0 SUMMARY

Unintended writes to flash memory can, at a minimum, cause it to output data that the system does not expect, forcing system reset or power sequencing to restore normal operation. Depending on the specific data written to the device, and the V_{pp} voltage at the time of the write, actual "permanent" alteration of flash memory contents can result from unintended program or erase. However, Intel flash memory, in combination with proper system interfacing techniques, easily eliminates the potential for either of these scenarios.

Closely analyze the powerup/down and reset behaviour of the system CPU and any interface logic that interacts with the flash memory. In the vast majority of cases, no problems will be found. If potential for unwanted writes does exist, however, nonvolatile/rewriteable memory protection can easily be included if incorporated early in the design, by following the hints described in this application note.



AB-29

APPLICATION BRIEF

Flash Memory Applications in Laser Printers

2

BRIAN DIPERT
MCD MARKETING APPLICATIONS

August 1993

Order Number: 292110-001

2-49

Flash Memory Applications in Laser Printers

CONTENTS

PAGE

1.0 INTRODUCTION 2-51

2.0 MEMORY USAGE IN LASER PRINTERS

2-52

2.1 System Code Memory 2-52

2.2 Font Storage 2-54

2.3 Image Storage and
Manipulation 2-55

CONTENTS

PAGE

3.0 SUMMARY 2-55

ADDITIONAL INFORMATION 2-55

Recently, several laser printer and printer peripheral companies have introduced products that incorporate flash memory. Their advertisements validate the unique capabilities and benefits that flash memory features provide. OEM interest, and predictions by market analysts like BIS Strategic Decisions, point to increasing future flash memory usage as laser printer manufacturers continue to differentiate their product lines to meet user needs. This application brief discusses the uses and benefits of flash memory in laser printer designs. Specifically, flash memory usage for system code storage, and for font and "font-like" data storage, will be highlighted.

1.0 INTRODUCTION

Within the computer industry, the laser printer market is one of the most rapidly growing business sectors. The graph of Figure 1 shows growth rate of various market

segments since 1988, as well as predicted growth through 1996. Laser printer proliferations are expanding to capture the needs of more and more user groups. Simultaneously, more and more computer users are turning to laser printers versus traditional "impact printer" alternatives, as features proliferate, capabilities expand, offices become more automated, and unit prices fall. This combination grows the total number of laser printer market segments, as well as the size of each segment.

When a large and steadily increasing supply of potential laser printer users exists, the "invisible hands" of economics unequivocally dictate that a large number of suppliers will appear to service this demand. What does this all mean to you, the laser printer manufacturer/designer? In a word, **competition!** How can any one company expand (or at a minimum maintain) their market share over the efforts of all others?

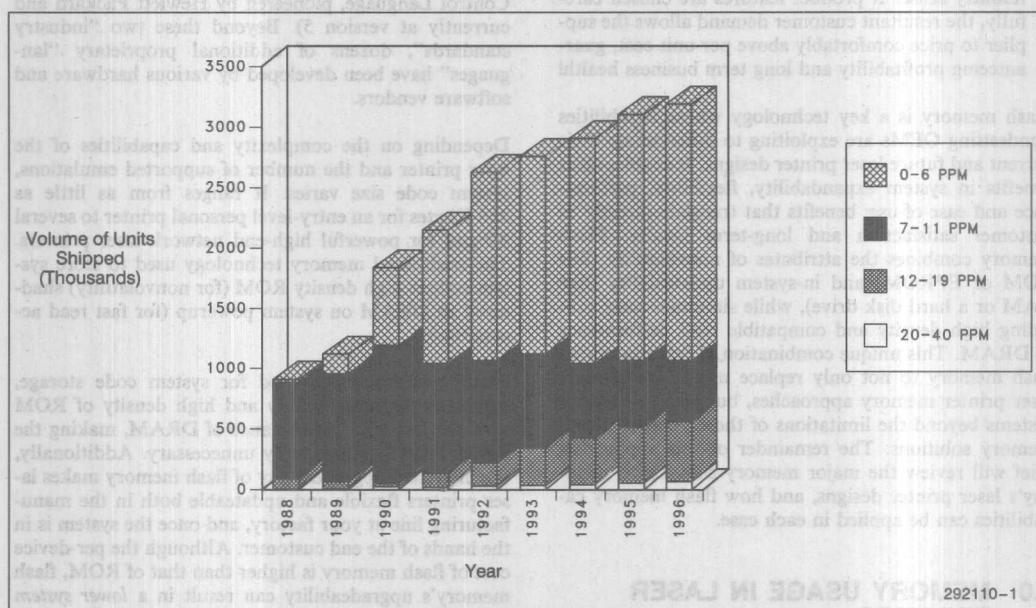


Figure 1. Laser Printer U.S. Market Growth 1988-1996
(BIS Strategic Decisions, 1992)

One way to do this is by providing equivalent product features as all other suppliers, at a lower unit price. This is referred to as **pricing-driven demand**. Unfortunately, as in the example of the "clone" market in today's PC industry, pricing-driven demand does not often translate to long-term financial health for your company.

The other method of establishing a leadership image (and the preferable means) is by stimulating **differentiation-based demand** through key product features that uniquely answer market needs. The advantages to this approach are many:

- Product differentiation makes good business **market share** sense. Uniquely meeting customer needs through product features establishes a short-term leadership image and the potential for a long-term protectable market position.
- Value-based pricing also makes good business **profitability** sense. If product features are chosen carefully, the resultant customer demand allows the supplier to price comfortably above per-unit cost, guaranteeing profitability and long term business health!

Flash memory is a key technology whose capabilities trendsetting OEMs are exploiting to differentiate their current and future laser printer designs. It enables clear benefits in system expandability, flexibility, performance and ease-of-use; benefits that translate directly to customer satisfaction and long-term loyalty. Flash memory combines the attributes of nonvolatility (like ROM or EPROM) and in-system updateability (like RAM or a hard disk drive), while simultaneously providing high density and compatible read performance to DRAM. This unique combination of features allows flash memory to not only replace more "traditional" laser printer memory approaches, but also to enhance systems beyond the limitations of these "conservative" memory solutions. The remainder of this application brief will review the major memory subsystems in today's laser printer designs, and how flash memory capabilities can be applied in each case.

2.0 MEMORY USAGE IN LASER PRINTERS

A high-level laser printer block diagram is shown in Figure 2. Memory uses in laser printers can be grouped in the following three areas:

- System and PDL (printer description language) emulation code storage
- Font (and font-like data type) storage
- Temporary bitmap image storage and manipulation

Memory is used differently in each case: therefore optimum memory features are similarly specialized, even within a common memory technology. Semiconductor

vendors, for example, have optimized various types of DRAMs for the applications in which they'll be used. Similarly, Intel has optimized unique product "families" within its flash memory line to match the requirements of applications like those found in laser printers. These product "families" will be referred to in the discussions below.

2.1 System Code Memory

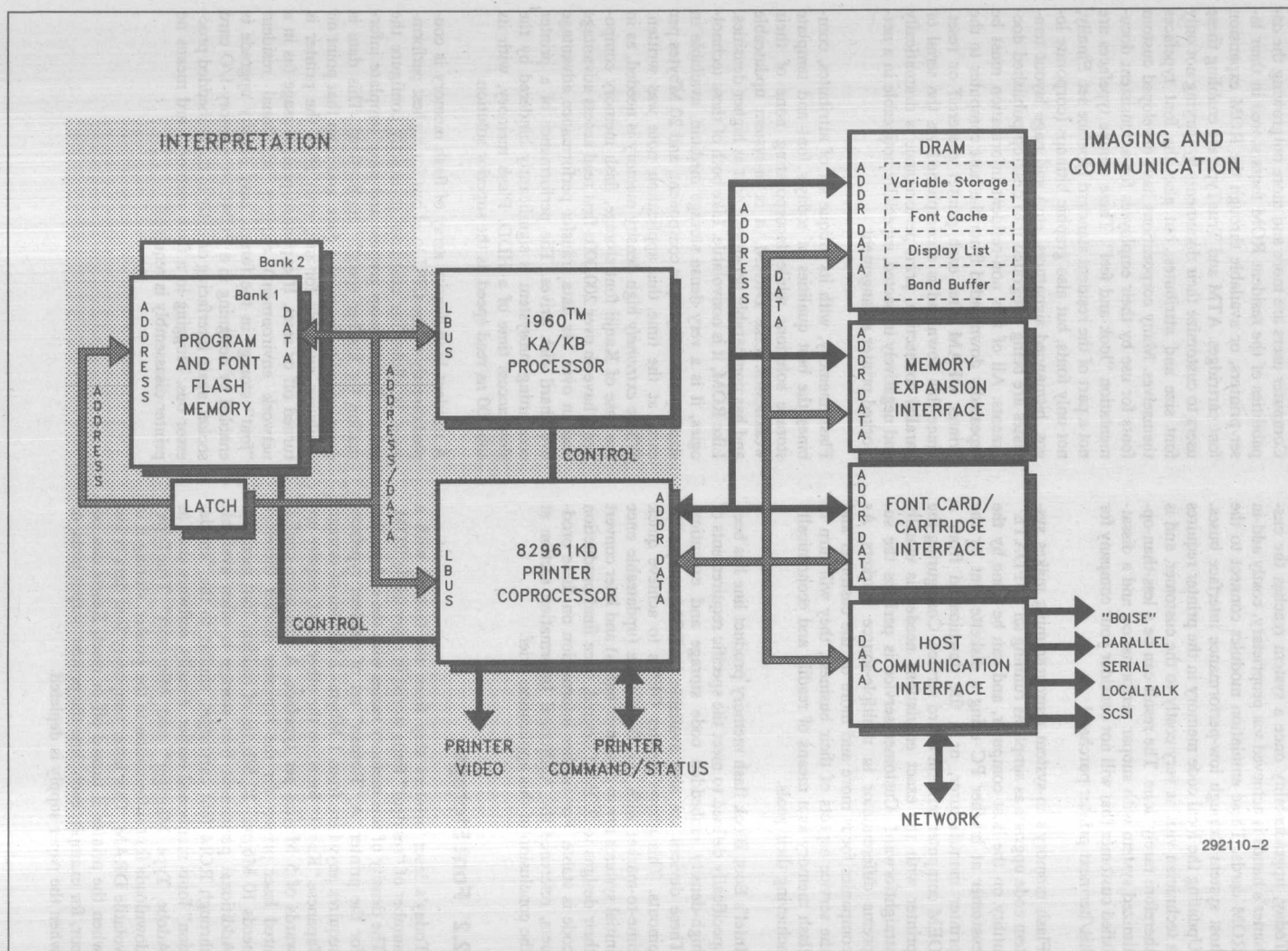
The system code memory stores the software that is executed by the embedded processor to run and control the laser printer. This includes code to interface the processor with the input/output, coprocessor, print drum and motor subsystems. System code memory also includes a large amount of software devoted to emulating various printer description languages. The most commonly known PDLs are PostScript® (pioneered by Adobe and currently at version 2.0) and PCL* (Printer Control Language, pioneered by Hewlett Packard and currently at version 5). Beyond these two "industry standards", dozens of additional proprietary "languages" have been developed by various hardware and software vendors.

Depending on the complexity and capabilities of the laser printer and the number of supported emulations, system code size varies. It ranges from as little as 256 Kbytes for an entry-level personal printer to several Mbytes for powerful high-end network laser printers. The traditional memory technology used to store system code is high density ROM (for nonvolatility) shadowed to DRAM on system powerup (for fast read access time).

Flash memory, when used for system code storage, combines the nonvolatility and high density of ROM with the fast read performance of DRAM, making the ROM/DRAM redundancy unnecessary. Additionally, the in-system upgradeability of flash memory makes laser printers flexible and updateable both in the manufacturing line at your factory, and once the system is in the hands of the end customer. Although the per-device cost of flash memory is higher than that of ROM, flash memory's upgradeability can result in a *lower system cost* through a laser printer's operating lifetime.

Flash memory eliminates costly inventory of ROMs, ROMs that must be scrapped if an enhanced software version is released or a software "bug" is discovered. Using flash memory, one hardware design can service multiple markets via simple "end-of-line" programming as the system leaves the factory. Additionally, diagnostic code can be programmed on the assembly line for full system debug, and replaced with the final software version later in the manufacturing flow.

Figure 2. Flash Memory, A Key Element In Leading Edge Laser Printers



Today, upgradeability once a system reaches the customer's hands is achieved via proprietary, costly add-in ROM cards. These emulation modules connect to the host system through low-performance interface buses. Updating the local code memory in the printer requires a technician visit, is very costly to the customer, and is therefore rarely done. The result can be a less-than-optimized system with subpar performance, and a dissatisfied customer that will not consider your company for his/her next printer purchase!

Flash memory's in-system reprogramming makes system code upgrade as simple as running an "UPDATE" utility on the host computer, and can be done by the customer at his/her PC using a diskette sent by the printer manufacturer, or a file downloaded from a OEM computer bulletin board service. Configuring the printer with the exact emulations needed is equally straightforward. Customer service is perhaps the supreme differentiator in multiple-source markets. As companies focus more and more on the customer and the service aspects of their business, they will turn to flash memory as a means of readily and economically achieving their goals.

Intel's Boot Block flash memory product line has been specifically defined to meet the specific requirements of high-density embedded code storage and execution. These devices are also available in ROM-compatible pinouts. This allows printer OEMs to achieve quick time-to-market with rev. 0 software (updateable once initial systems are in customer hands) and later convert their designs to ROM if desired, once final production code is stable. For further information on these products, reference the Additional Information section at the conclusion of this application brief.

2.2 Font Storage

Today's laser printers ship from the factory with a number of "resident" fonts stored in nonvolatile ROM. The density of this memory varies with the end market for the printer. A "Roman" set of resident typefaces requires anywhere from 1 Mbyte–2 Mbytes of storage. Japanese "Kanji" fonts, on the other hand, require upwards of 5 Mbytes per typeface. A minimum-configured laser printer for the Japanese market therefore needs 10 Mbytes–20 Mbytes of resident font memory. Additional permanent font storage is often available through ROM font cartridges, similar to the "emulation" fonts mentioned earlier. Finally, software such as Adobe Type Manager® and Microsoft® TrueType® downloads font information to the printer, storing it in volatile DRAM. This latter temporary font data is lost when the printer is turned off or reset. Resets can occur, for example, each time the printer output jams, or when the paper supply is depleted.

Computer users are more and more outgrowing the capabilities of the resident ROM fonts stored in their laser printers, or available through the ROM expansion font cartridges. ATM and TrueType are enabling these users to customize their documents by varying not only font size and attributes, but also the font typefaces themselves. Many corporations have developed custom fonts for use by their employees for a consistent documentation "look and feel". These unique typefaces are not a part of the resident standard typeface set. Finally, not only fonts, but also graphic bitmaps (corporate logos, bitmapped signatures, etc.) and page layout templates are being integrated into desktop-published documents. All of this non-resident information must be repeatedly downloaded from the host computer to the printer DRAM after each printer poweroff or reset. Since this download is accomplished via the serial or parallel connection, print performance is dramatically and negatively impacted, especially noticeable in a networked printer arrangement.

Flash memory, with its unique set of attributes, combines the best qualities of today's font and template storage solutions while incorporating none of their weaknesses. Like DRAM, it is in-system updateable and has comparable per-device cost at higher densities. Like ROM, it is nonvolatile. Like both of these technologies, it is a very dense storage medium, available in sizes up to 1 Mbyte per component, and 20 Mbytes per card, at the time this application note was written. Where *extremely* high density memory is needed, as in the case of Kanji font storage, flash memory components have an over 200,000x first read access advantage and an over 14x data transfer performance advantage over hard disk drives. The performance of a printer computing subsystem is significantly hindered by the slow access time of a HDD. Flash memory, with its sub-100 ns read speed, is the superior solution.

A resident high-density array of flash memory is coupled directly to the CPU local bus for highest performance. It allows the customer to exactly configure the printer font, bitmap graphic and page template information for his/her specific applications. This data is downloaded to the printer *once*, and from that point on is always available for use, even after the printer is turned off or reset. If expanded printer usage (as in a network environment) requires additional resident "font" storage in the future, easy density upgrade is enabled by designing in a PCMCIA memory-I/O card socket, again interfacing directly to the embedded processor bus. Plugging in a flash memory card means no printer disassembly is required!

Intel's FlashFile™ flash memory component and Series 2 flash memory card lines combine the high density and high performance required for resident "font" storage. For further information on these products, reference the Additional Information section at the conclusion of this application brief.

2.3 Image Storage and Manipulation

The temporary graphic memory subsystem stores the image to be printed as it is "constructed" by the processor from data provided by the host computer. Optimum characteristics of this memory include full "real-time" bit-level alteration, infinite rewrite capability and fast read/write performance. Nonvolatility is not required in this area of the memory subsystem. Therefore, DRAM will continue to be the memory of choice for temporary image storage.

3.0 SUMMARY

This application brief has discussed the various memory subsystems in today's laser printers, and their operating characteristics. Flash memory is an exciting new approach that offers the very real potential to significantly improve your next-generation laser printer designs. Its capabilities are superior to traditional solutions in the system code and font memory areas, and enable laser printers that are more expandable, more flexible, higher performance and easier to use than ever before. The end result is a satisfied customer, a customer that will choose your product over a competitor's, and a customer that will remain loyal to your company far into the future.

2

ADDITIONAL INFORMATION

For additional information on the Intel flash memory products mentioned in this article, please reference the following documents, available through your local Intel sales representative.

Boot Block Components

28F001BX Datasheet	
28F200BX/28F002BX Datasheet	
28F400BX/28F004BX Datasheet	
ER-26 "The Intel 28F001BX-T and 28F001BX-B Flash Memories"	
ER-29 "The Intel 2/4 Mbit Boot Block Flash Memory Family"	

Order Number

290406
290448
290451
294010
294013

FlashFile Components

28F008SA Datasheet	
AP-359 "28F008SA Hardware Interfacing"	
AP-360 "28F008SA Software Drivers"	
AP-364 "28F008SA Automation and Algorithms"	
ER-27 "The Intel 28F008SA Flash Memory"	

Order Number

290429
292094
292095
292099
294011

FlashFile Series 2 Cards

Series 2 Flash Memory Card Datasheet	
AP-361 "Implementing the Integrated Registers of the Series 2 Flash Memory Card"	

Order Number

290434
292096

General Flash Information

AP-357 "Power Supply Solutions for Flash Memory"	
ER-20 "ETOX II Flash Memory Technology"	
ER-28 "ETOX III Flash Memory Technology"	

Order Number

292092
294005
294012

*Microsoft and TrueType are trademarks of Microsoft Corporation. Adobe Type Manager and PostScript are trademarks of Adobe Systems Incorporated. PCL is a trademark of Hewlett Packard Corporation. ETOX and FlashFile are trademarks of Intel Corporation.



FlashFile™ Components

3

25

DD28F032SA

32-MBIT (2 MBIT x 16, 4 MBIT x 8) FLASHFILE™ MEMORY

- **User-Selectable 3.3V or 5V V_{CC}**
- **User-Configurable x8 or x16 Operation**
- **70 ns Maximum Access Time**
- **0.43 MB/sec Write Transfer Rate**
- **1 Million Erase Cycles per Block**
- **56-Lead, 1.2 x 14 x 20mm Dual Die Advanced TSOP Package Technology**
- **Revolutionary Architecture**
 - 100% Backwards Compatible with Intel 28F016SA
 - Pipelined Command Execution
 - Write during Erase
- **2 mA Typical I_{CC} in Static Mode**
- **2 μ A Typical Deep Power-Down**
- **64 Independently Lockable Blocks**
- **State-of-the-Art 0.6 μ m ETOX IV Flash Technology**

Intel's DD28F032SA 32-Mbit FlashFile™ Memory is a revolutionary architecture which enables the design of truly mobile, high performance, personal computing and communication products. With innovative capabilities, low power operation and very high read/write performance, the DD28F032SA is also the ideal choice for designing embedded mass storage flash memory systems.

The DD28F032SA is the result of highly advanced packaging innovation which encapsulates two 28F016SA die in a single Dual Die Thin Small Outline Package (DDTSOP).

The DD28F032SA is the highest density, highest performance non-volatile read/write solution for solid-state storage applications. Its symmetrically blocked architecture (100% compatible with the 28F016SA 16-Mbit FlashFile memory), very high cycling, low power 3.3V operation, very fast write and read performance and selective block locking provide a highly flexible memory component suitable for high density memory cards, Resident Flash Arrays and PCMCIA-ATA Flash Drives. The DD28F032SA's dual read voltage enables the design of memory cards which can interchangeably be read/written in 3.3V and 5.0V systems. Its x8/x16 architecture allows the optimization of memory to processor interface. The flexible block locking option enables bundling of executable application software in a Resident Flash Array or memory card. The DD28F032SA will be manufactured on Intel's 0.6 μ m ETOX IV technology.

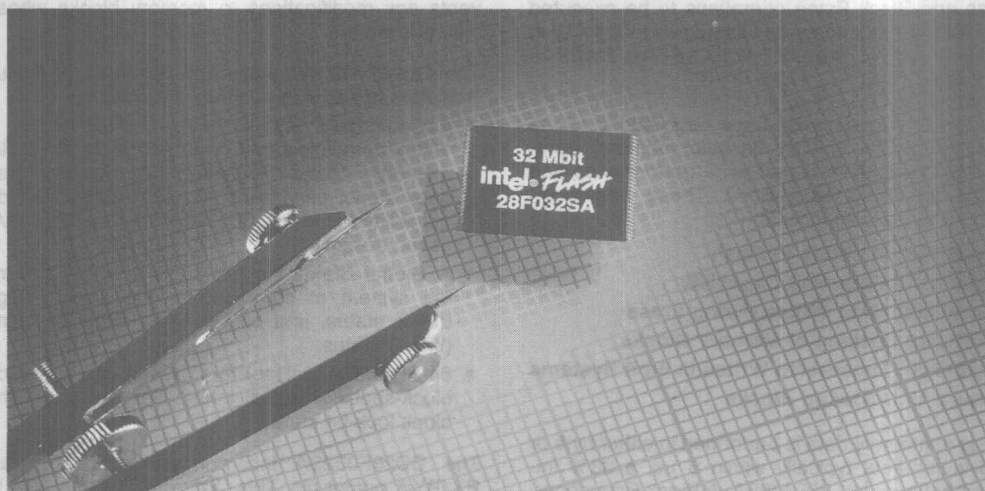


Figure 1

290490-2

1.0 PRODUCT OVERVIEW

The DD28F032SA is a high performance 32-Mbit (33,554,432-bit) block erasable non-volatile random access memory organized as either 2 Mword x 16 or 4 Mbyte x 8. The DD28F032SA is built using two 28F016SA chips encapsulated in a single 56L-TSOP type I package. The DD28F032SA includes sixty-four 64-KB (65,536) blocks or sixty-four 32-KW (32,768) blocks.

The implementation of a new architecture, with many enhanced features, will improve the device operating characteristics and results in greater product reliability and ease of use.

Among the significant enhancements on the DD28F032SA:

- 3.3V Low Power Capability
- Improved Write Performance
- Dedicated Block Write/Erase Protection

A 3/5# input pin reconfigures the device internally for optimized 3.3V or 5.0V read/write operation.

The DD28F032SA will be available in a 56-lead, 1.2mm thick, 14mm x 20mm Dual Die TSOP type I package. This form factor and pinout allow for very high board layout densities. The DD28F032SA is pinout and footprint compatible with the 28F016SA.

A Command User Interface (CUI) serves as the system interface between the microprocessor or microcontroller and the internal memory operation.

Internal Algorithm Automation allows Word/Byte Writes and Block Erase operations to be executed using a Two-Write command sequence to the CUI in the same way as the 28F016SA 16-Mbit FlashFile memory.

A Superset of commands have been added to the basic 28F008SA (8-Mbit flashfile memory) command-set to achieve higher write performance and provide additional capabilities. These new commands and features include:

- **Page Buffer Writes to Flash**
- **Command Queueing Capability**
- **Automatic Data Writes during Erase**
- **Software Locking of Memory Blocks**
- **Two-Byte Successive Writes in 8-bit Systems**
- **Erase All Unlocked Blocks**

Writing of memory data is performed in either byte or word increments typically within 6 μ s, a 33% improvement over the 28F008SA. A Block Erase operation erases one of the 64 blocks in typically 0.6 sec, independent of the other blocks, which is a 65% improvement over the 28F008SA.

Each block can be written and erased a minimum of 100,000 cycles. Systems can achieve 1 million Block Erase Cycles by providing wear-leveling algorithms and graceful block retirement. These techniques have already been employed in many flash file systems and in Hard Disk Drive designs.

The DD28F032SA incorporates four Page Buffers of 256 Bytes (128 Words) each to allow page data writes. This feature can improve a system write performance by up to 4.8 times over previous flash memory devices.

All operations are started by a sequence of Write commands to the device. Three Status Registers (described in detail later) and a RY/BY# output pin provide information on the progress of the requested operation.

The DD28F032SA allows queueing of the next operation while the memory executes the current operation. This eliminates system overhead when writing several bytes in a row to the array or erasing several blocks at the same time. The DD28F032SA can also perform write operations to one block of memory while performing erase of another block. However, simultaneous write and/or erase operations are not allowed on both 28F016SA devices. See modes of operation in Section 3.0.

The DD28F032SA provides user-selectable block locking to protect code or data such as Device Drivers, PCMCIA card information, ROM-Executable O/S or Application code. Each block has an associated non-volatile lock-bit which determines the lock status of the block. In addition, the DD28F032SA has a master Write Protect pin (WP#) which prevents any modifications to memory blocks whose lock-bits are set.

The DD28F032SA contains three types of Status Registers to accomplish various functions:

- A Compatible Status Register (CSR) which is 100% compatible with the 28F008SA FlashFile memory's Status Register. This register, when used alone, provides a straightforward upgrade capability to the DD28F032SA from a 28F008SA-based design.
- A Global Status Register (GSR) which informs the system of command Queue status, Page Buffer status, and overall Write State Machine (WSM) status.
- 64 Block Status Registers (BSRs) which provide block-specific status information such as the block lock-bit status.

The DD28F032SA incorporates an open drain RY/BY# output pin. This feature allows the user to OR-tie many RY/BY# pins together in a multiple memory configuration such as a Resident Flash Array.

Other configurations of the RY/BY# pin are enabled via special CUI commands.

Consult the 28F016SA user's manual for more detail.

The DD28F032SA incorporates three chip enable input pins: CE₀#, CE₁# and CE₂#. The active low combination of CE₀# and CE₁# controls the lower 28F016SA. The active low combination of CE₀# and CE₂# controls the upper 28F016SA.

The BYTE# pin allows either x8 or x16 read/writes to the DD28F032SA. BYTE# at logic low selects 8-bit mode with address A₀ selecting between low byte and high byte. On the other hand, BYTE# at logic high enables 16-bit operation with address A₁ becoming the lowest order address and address A₀ is not used (don't care).

The DD28F032SA incorporates an Automatic Power Saving (APS) feature which substantially reduces

the active current when the device is in static mode of operation (addresses not switching).

A Deep Power-Down mode of operation is invoked when the RP# (called PWD on the 28F008SA) pin transitions low. This mode provides additional write protection by acting as a device reset pin during power transitions. In the Deep Power-Down state, the WSM is reset (any current operation will abort) and the CSR, GSR and BSR registers are cleared.

A CMOS Standby mode of operation is enabled when either CE₀# or both CE₁# and CE₂# transition high and RP# stays high with all input control pins at CMOS levels.

2.0 DEVICE PINOUT

The DD28F032SA 56L-Dual Die TSOP Type I pinout configuration is shown in Figure 1.

3

2.1 Lead Descriptions

Symbol	Type	Name and Function
A ₀	INPUT	BYTE-SELECT ADDRESS: Selects between high and low byte when device is in x8 mode. This address is latched in x8 Data Writes. Not used in x16 mode (i.e., the A ₀ input buffer is turned off when BYTE# is high).
A ₁ –A ₁₅	INPUT	WORD-SELECT ADDRESSES: Select a word within one 64-KByte block. A ₆ –A ₁₅ selects 1 of 1024 rows, and A ₁ –A ₅ selects 16 of 512 columns. These addresses are latched during Data Writes.
A ₁₆ –A ₂₀	INPUT	BLOCK-SELECT ADDRESSES: Select 1 of 32 Erase blocks. These addresses are latched during Data Writes, Erase and Lock-Block operations.
DQ ₀ –DQ ₇	INPUT/OUTPUT	LOW-BYTE DATA BUS: Inputs data and commands during CUI write cycles. Outputs array, buffer, identifier or status data in the appropriate Read mode. Floated when the chip is de-selected or the outputs are disabled.
DQ ₈ –DQ ₁₅	INPUT/OUTPUT	HIGH-BYTE DATA BUS: Inputs data during x16 Data-Write operations. Outputs array, buffer or identifier data in the appropriate Read mode; not used for Status register reads. Floated when the chip is de-selected or the outputs are disabled.
CE ₀ #, CE ₁ #, CE ₂ #	INPUT	CHIP ENABLE INPUTS: Activate the device's control logic, input buffers, decoders and sense amplifiers. CE ₀ #, CE ₁ # disable/enable Lower 28F016SA while CE ₀ #, CE ₂ # disable/enable Upper 28F016SA. CE ₀ # active low enables chip operation while CE ₁ # or CE ₂ # select between the lower and upper 2 MBytes. CE ₁ # and CE ₂ # must not be active low simultaneously.
RP#	INPUT	RESET/POWER-DOWN: RP# low places the device in a Deep Power-Down state. All circuits that burn static power, even those circuits enabled in standby mode, are turned off. When returning from Deep Power-Down, a recovery time is required to allow these circuits to power-up. When RP# goes low, any current or pending WSM operation(s) are terminated, and the device is reset. All Status registers return to ready (with all status flags cleared).

		OUTPUT ENABLE: Gates device data through the output buffers when low. The outputs float to tri-state off when OE# is high. NOTE: CE _X # overrides OE#, and OE# overrides WE#.
WE#	INPUT	WRITE ENABLE: Controls access to the CUI, Page Buffers, Data Queue Registers and Address Queue Latches. WE# is active low, and latches both address and data (command or array) on its rising edge.
RY/BY#	OPEN DRAIN OUTPUT	READY/BUSY: Indicates status of the internal WSM. When low, it indicates that the WSM is busy performing an operation. RY/BY# high indicates that the WSM is ready for new operations (or WSM has completed all pending operations), or Erase is Suspended, or the device is in Deep Power-Down mode. This output is always active (i.e., not floated to tri-state off when OE# or CE ₀ #, CE ₁ # are high), except if a RY/BY# Pin Disable command is issued.
WP#	INPUT	WRITE PROTECT: Erase blocks can be locked by writing a non-volatile lock-bit for each block. When WP# is low, those locked blocks as reflected by the Block-Lock Status bits (BSR.6), are protected from Data Writes or Erases. When WP# is high, all blocks can be Written or Erased regardless of the state of the lock-bits. The WP# input buffer is disabled when RP# transitions low (Deep Power-Down mode).
BYTE#	INPUT	BYTE ENABLE: BYTE# low places device in x8 mode. All data is then input or output on DQ ₀ –DQ ₇ , and DQ ₈ –DQ ₁₅ float. Address A ₀ selects between the high and low byte. BYTE# high places the device in x16 mode, and turns off the A ₀ input buffer. Address A ₁ , then becomes the lowest order address.
3/5#	INPUT	3.3V/5.0V SELECT: 3/5# high configures internal circuits for 3.3V operation. 3/5# low configures internal circuits for 5.0V operation. NOTES: Reading the array with 3/5# high in a 5.0V system could damage the device. There is a significant delay from 3/5# switching to access valid data.
V _{PP}	SUPPLY	ERASE/WRITE POWER SUPPLY: For erasing memory array blocks or writing words/bytes/pages into the flash array.
V _{CC}	SUPPLY	DEVICE POWER SUPPLY (3.3V ± 0.3V, 5.0V ± 0.5V): Do not leave any power pins floating.
GND	SUPPLY	GROUND FOR ALL INTERNAL CIRCUITRY: Do not leave any ground pins floating.
NC		NO CONNECT: No internal connection to die, lead may be driven or left floating

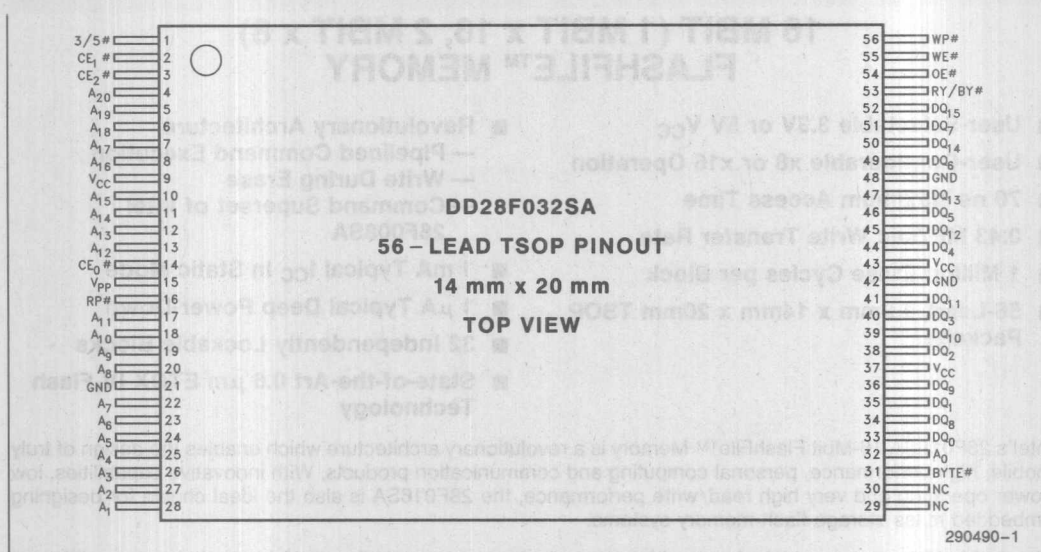


Figure 2. TSOP Pinout Configuration

3.0 MODES OF OPERATION

RP#	CE ₀ #	CE ₁ #	CE ₂ #	Lower 28F016SA	Upper 28F016SA	DD28F032SA Chip
0	X	X	X	DPD	DPD	DPD
1	1	X	X	Standby	Standby	Standby
1	0	0	1	Active	Standby	Active
1	0	1	0	Standby	Active	Active
1	0	1	1	Standby	Standby	Standby
1	0	0	0	Illegal Condition		

X = Don't Care

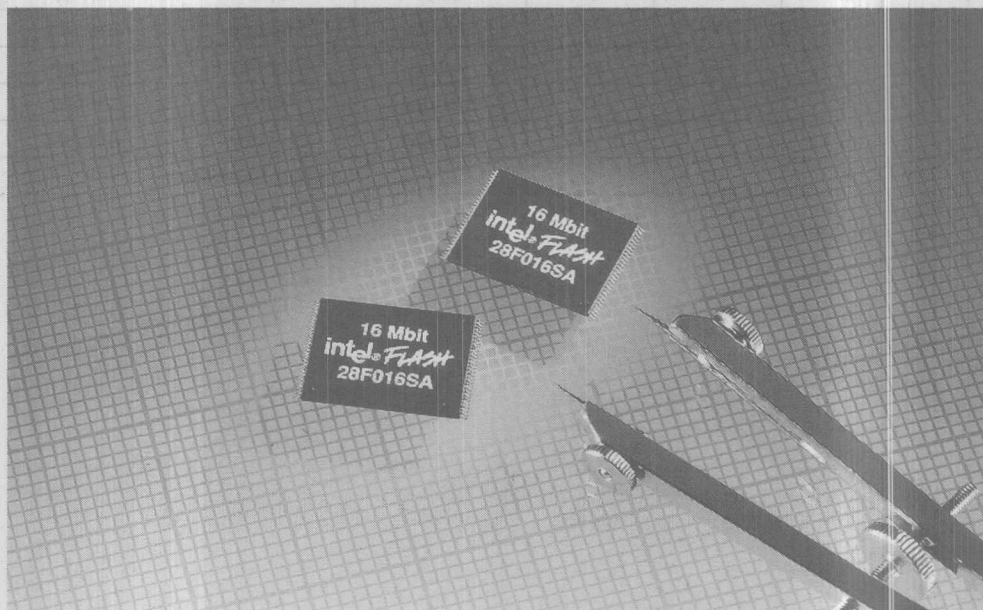
DPD = Deep Power-Down

28F016SA 16 MBIT (1 MBIT x 16, 2 MBIT x 8) FLASHFILE™ MEMORY

- User-Selectable 3.3V or 5V V_{CC}
- User-Configurable x8 or x16 Operation
- 70 ns Maximum Access Time
- 0.43 MB/sec Write Transfer Rate
- 1 Million Erase Cycles per Block
- 56-Lead, 1.2mm x 14mm x 20mm TSOP Package
- Revolutionary Architecture
 - Pipelined Command Execution
 - Write During Erase
 - Command Superset of Intel 28F008SA
- 1 mA Typical I_{CC} in Static Mode
- 1 μA Typical Deep Power-Down
- 32 Independently Lockable Blocks
- State-of-the-Art 0.6 μm ETOX IV Flash Technology

Intel's 28F016SA 16-Mbit FlashFile™ Memory is a revolutionary architecture which enables the design of truly mobile, high performance, personal computing and communication products. With innovative capabilities, low power operation and very high read/write performance, the 28F016SA is also the ideal choice for designing embedded mass storage flash memory systems.

The 28F016SA is a very high density, highest performance non-volatile read/write solution for solid-state storage applications. Its symmetrically blocked architecture (100% compatible with the 28F008SA 8-Mbit FlashFile™ memory), extended cycling, low power 3.3V operation, very fast write and read performance and selective block locking provide a highly flexible memory component suitable for high density memory cards, Resident Flash Arrays and PCMCIA-ATA Flash Drives. The 28F016SA's dual read voltage enables the design of memory cards which can interchangeably be read/written in 3.3V and 5.0V systems. Its x8/x16 architecture allows the optimization of memory to processor interface. The flexible block locking option enables bundling of executable application software in a Resident Flash Array or memory card. Manufactured on Intel's 0.6 μm ETOXIV process technology, the 28F016SA is the most cost-effective, high-density 3.3V flashfile memory.



290489-1

1.0 INTRODUCTION

The documentation of the Intel 28F016SA memory device includes this data sheet, a detailed user's manual, and a number of application notes, all of which are referenced at the end of this data sheet.

The data sheet is intended to give an overview of the chip feature-set and of the operating AC/DC specifications. The 28F016SA User's Manual provides complete descriptions of the user modes, system interface examples and detailed descriptions of all principles of operation. It also contains the full list of software algorithm flowcharts, and a brief section on compatibility with Intel 28F008SA.

1.1 Product Overview

The 28F016SA is a high performance 16 Mbit (16,777,216 bit) block erasable non-volatile random access memory organized as either 1 Mword x 16 or 2 Mbyte x 8. The 28F016SA includes thirty-two 64 KB (65,536) blocks or thirty-two 32-KW (32,768) blocks. A chip memory map is shown in Figure 3.

The implementation of a new architecture, with many enhanced features, will improve the device operating characteristics and results in greater product reliability and ease of use.

Among the significant enhancements on the 28F016SA:

- 3.3V Low Power Capability
- Improved Write Performance
- Dedicated Block Write/Erase Protection

A 3/5# input pin reconfigures the device internally for optimized 3.3V or 5.0V read/write operation.

The 28F016SA will be available in a 56-lead, 1.2mm thick, 14mm x 20mm TSOP type 1 package. This form factor and pinout allow for very high board layout densities.

A Command User Interface (CUI) serves as the system interface between the microprocessor or micro-controller and the internal memory operation.

Internal Algorithm Automation allows Byte/Word Writes and Block Erase operations to be executed using a Two-Write command sequence to the CUI in the same way as the 28F008SA 8-Mbit FlashFile™ memory.

A Superset of commands have been added to the basic 28F008SA command-set to achieve higher write performance and provide additional capabilities. These new commands and features include:

- Page Buffer Writes to Flash
- Command Queuing Capability
- Automatic Data Writes During Erase
- Software Locking of Memory Blocks
- Two-Byte Successive Writes in 8-bit Systems
- Erase All Unlocked Blocks

Writing of memory data is performed in either byte or word increments typically within 6 μ sec, a 33% improvement over the 28F008SA. A Block Erase operation erases one of the 32 blocks in typically 0.6 sec, independent of the other blocks, which is about 65% improvement over the 28F008SA.

Each block can be written and erased a minimum of 100,000 cycles. Systems can achieve 1 million Block Erase Cycles by providing wear-leveling algorithms and graceful block retirement. These techniques have already been employed in many flash file systems and Hard Disk Drive designs.

The 28F016SA incorporates two Page Buffers of 256 Bytes (128 Words) each to allow page data writes. This feature can improve a system write performance by up to 4.8 times over previous flash memory devices.

All operations are started by a sequence of Write commands to the device. Three Status Registers (described in detail later) and a RY/BY# output pin provide information on the progress of the requested operation.

While the 28F008SA requires an operation to complete before the next operation can be requested, the 28F016SA allows queuing of the next operation while the memory executes the current operation. This eliminates system overhead when writing several bytes in a row to the array or erasing several blocks at the same time. The 28F016SA can also perform write operations to one block of memory while performing erase of another block.

The 28F016SA provides user-selectable block locking to protect code or data such as Device Drivers, PCMCIA card information, ROM-Executable O/S or Application Code. Each block has an associated non-volatile lock-bit which determines the lock status of the block. In addition, the 28F016SA has a master Write Protect pin (WP#) which prevents any modifications to memory blocks whose lock-bits are set.

The 28F016SA contains three types of Status Registers to accomplish various functions:

- A Compatible Status Register (CSR) which is 100% compatible with the 28F008SA FlashFile memory's Status Register. This register, when used alone, provides a straightforward upgrade capability to the 28F016SA from a 28F008SA-based design.
- A Global Status Register (GSR) which informs the system of command Queue status, Page Buffer status, and overall Write State Machine (WSM) status.
- 32 Block Status Registers (BSRs) which provide block-specific status information such as the block lock-bit status.

The GSR and BSR memory maps for Byte-Wide and Word-Wide modes are shown in Figures 4.1 and 4.2.

The 28F016SA incorporates an open drain RY/BY# output pin. This feature allows the user to OR-tie many RY/BY# pins together in a multiple memory configuration such as a Resident Flash Array.

Other configurations of the RY/BY# pin are enabled via special CUI commands and are described in detail in the 28F016SA User's Manual.

The 28F016SA also incorporates a dual chip-enable function with two input pins, CE₀# and CE₁#. These pins have exactly the same functionality as the regular chip-enable pin CE# on the 28F008SA. For minimum chip designs, CE₁# may be tied to ground and use CE₀# as the chip enable input. The 28F016SA uses the logical combination of these two signals to enable or disable the entire chip. Both CE₀# and CE₁# must be active low to enable the device and if either one becomes inactive, the chip will be disabled. This feature, along with the open drain RY/BY# pin, allows the system designer to reduce the number of control pins used in a large array of 16-Mbit devices.

The BYTE# pin allows either x8 or x16 read/writes to the 28F016SA. BYTE# at logic low selects 8-bit mode with address A₀ selecting between low byte

and high byte. On the other hand, BYTE# at logic high enables 16-bit operation with address A₁ becoming the lowest order address and address A₀ is not used (don't care). A device block diagram is shown in Figure 1.

The 28F016SA is specified for a maximum access time of 70 ns (t_{ACC}) at 5.0V operation (4.75V to 5.25V) over the commercial temperature range (0°C to +70°C). A corresponding maximum access time of 120 ns at 3.3V (3.0V to 3.6V and 0°C to +70°C) is achieved for reduced power consumption applications.

The 28F016SA incorporates an Automatic Power Saving (APS) feature which substantially reduces the active current when the device is in static mode of operation (addresses not switching).

In APS mode, the typical I_{CC} current is 1 mA at 5.0V (0.8 mA at 3.3V).

A Deep Power-Down mode of operation is invoked when the RP# (called $\overline{\text{PWD}}$ on the 28F008SA) pin transitions low. This mode brings the device power consumption to less than 1.0 μA , typically, and provides additional write protection by acting as a device reset pin during power transitions. A reset time of 400 ns is required from RP# switching high until outputs are again valid. In the Deep Power-Down state, the WSM is reset (any current operation will abort) and the CSR, GSR and BSR registers are cleared.

A CMOS Standby mode of operation is enabled when either CE₀# or CE₁# transitions high and RP# stays high with all input control pins at CMOS levels. In this mode, the device typically draws an I_{CC} standby current of 50 μA .

2.0 DEVICE PINOUT

The 28F016SA 56L-TSOP Type I pinout configuration is shown in Figure 2.

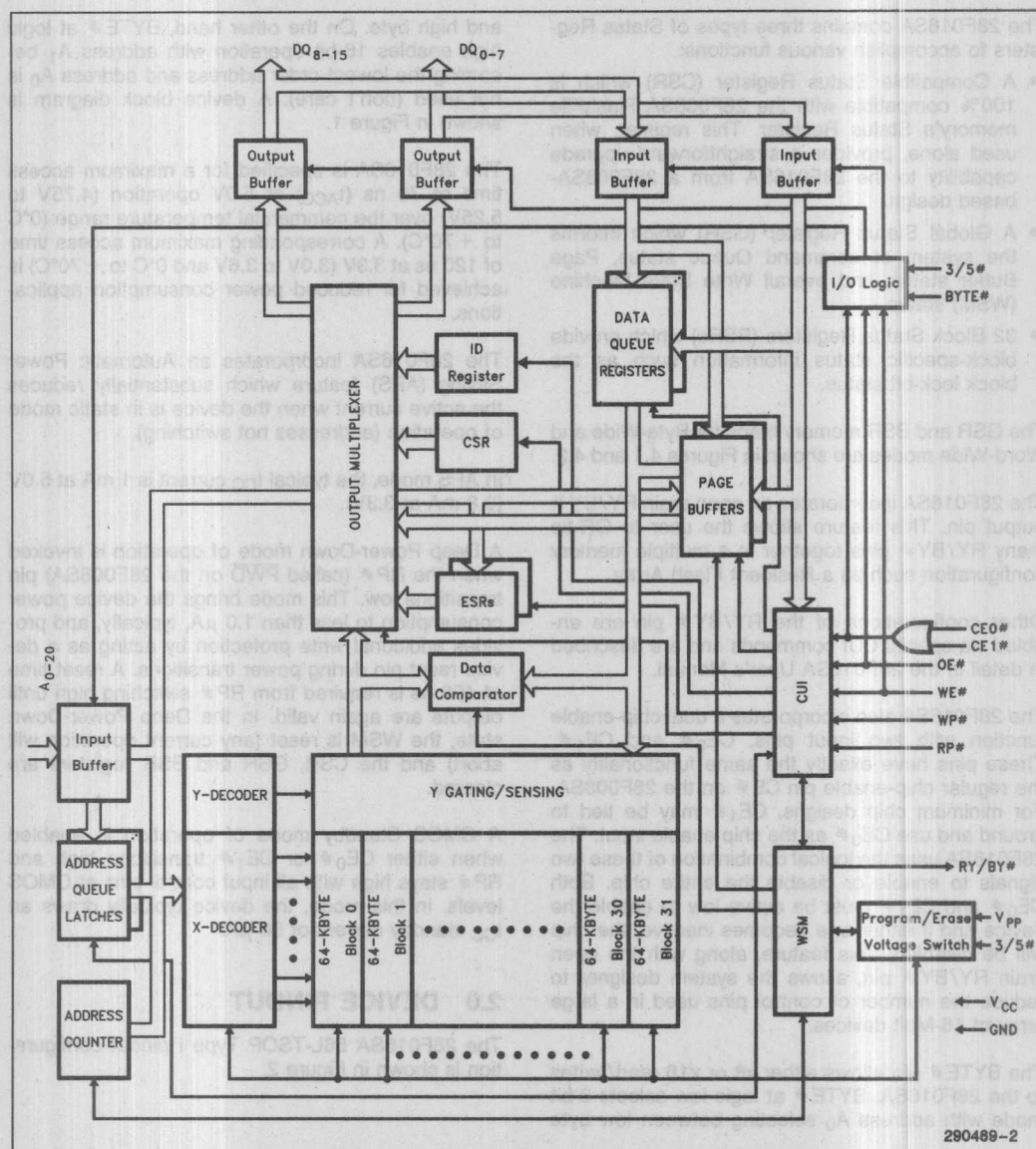


Figure 1. 28F016SA Block Diagram
Architectural Evolution Includes Page Buffers, Queue Registers and Extended Status Registers.

2.1 Lead Descriptions

Symbol	Type	Name and Function
A ₀	INPUT	BYTE-SELECT ADDRESS: Selects between high and low byte when device is in x8 mode. This address is latched in x8 Data Writes. Not used in x16 mode (i.e., the A ₀ input buffer is turned off when BYTE # is high).
A ₁ –A ₁₅	INPUT	WORD-SELECT ADDRESSES: Select a word within one 64-Kbyte block. A ₆ – ₁₅ selects 1 of 1024 rows, and A ₁ – ₅ selects 16 of 512 columns. These addresses are latched during Data Writes.
A ₁₆ –A ₂₀	INPUT	BLOCK-SELECT ADDRESSES: Select 1 of 32 Erase blocks. These addresses are latched during Data Writes, Erase and Lock-Block operations.
DQ ₀ –DQ ₇	INPUT/OUTPUT	LOW-BYTE DATA BUS: Inputs data and commands during CUI write cycles. Outputs array, buffer, identifier or status data in the appropriate Read mode. Floated when the chip is de-selected or the outputs are disabled.
DQ ₈ –DQ ₁₅	INPUT/OUTPUT	HIGH-BYTE DATA BUS: Inputs data during x16 Data-Write operations. Outputs array, buffer or identifier data in the appropriate Read mode; not used for Status register reads. Floated when the chip is de-selected or the outputs are disabled.
CE ₀ #, CE ₁ #	INPUT	CHIP ENABLE INPUTS: Activate the device's control logic, input buffers, decoders and sense amplifiers. With either CE ₀ # or CE ₁ # high, the device is de-selected and power consumption reduces to Standby levels upon completion of any current Data-Write or Erase operations. Both CE ₀ #, CE ₁ # must be low to select the device. All timing specifications are the same for both signals. Device Selection occurs with the latter falling edge of CE ₀ # or CE ₁ #. The first rising edge of CE ₀ # or CE ₁ # disables the device.
RP #	INPUT	RESET/POWER-DOWN: RP # low places the device in a Deep Power-Down state. All circuits that burn static power, even those circuits enabled in standby mode, are turned off. When returning from Deep Power-Down, a recovery time of 400 ns is required to allow these circuits to power-up. When RP # goes low, any current or pending WSM operation(s) are terminated, and the device is reset. All Status registers return to ready (with all status flags cleared).
OE #	INPUT	OUTPUT ENABLE: Gates device data through the output buffers when low. The outputs float to tri-state off when OE # is high. NOTE: CE _x # overrides OE #, and OE # overrides WE #.
WE #	INPUT	WRITE ENABLE: Controls access to the CUI, Page Buffers, Data Queue Registers and Address Queue Latches. WE # is active low, and latches both address and data (command or array) on its rising edge.
RY/BY #	OPEN DRAIN OUTPUT	READY/BUSY: Indicates status of the internal WSM. When low, it indicates that the WSM is busy performing an operation. RY/BY # high indicates that the WSM is ready for new operations (or WSM has completed all pending operations), or Erase is Suspended, or the device is in deep power-down mode. This output is always active (i.e., not floated to tri-state off when OE # or CE ₀ #, CE ₁ # are high), except if a RY/BY # Pin Disable command is issued.

Symbol	Type	Name and Function
WP#	INPUT	WRITE PROTECT: Erase blocks can be locked by writing a non-volatile lock-bit for each block. When WP# is low, those locked blocks as reflected by the Block-Lock Status bits (BSR.6), are protected from inadvertent Data Writes or Erases. When WP# is high, all blocks can be Written or Erased regardless of the state of the lock-bits. The WP# input buffer is disabled when RP# transitions low (deep power-down mode).
BYTE#	INPUT	BYTE ENABLE: BYTE# low places device in x8 mode. All data is then input or output on DQ ₀₋₇ , and DQ ₈₋₁₅ float. Address A ₀ selects between the high and low byte. BYTE# high places the device in x16 mode, and turns off the A ₀ input buffer. Address A ₁ , then becomes the lowest order address.
3/5#	INPUT	3.3/5.0 VOLT SELECT: 3/5# high configures internal circuits for 3.3V operation. 3/5# low configures internal circuits for 5.0V operation. NOTES: Reading the array with 3/5# high in a 5.0V system could damage the device. There is a significant delay from 3/5# switching to valid data.
V _{PP}	SUPPLY	ERASE/WRITE POWER SUPPLY: For erasing memory array blocks or writing words/bytes/pages into the flash array.
V _{CC}	SUPPLY	DEVICE POWER SUPPLY (3.3V ± 0.3V, 5.0V ± 0.5V): Do not leave any power pins floating.
GND	SUPPLY	GROUND FOR ALL INTERNAL CIRCUITRY: Do not leave any ground pins floating.
NC		NO CONNECT: No internal connection to die, lead may be driven or left floating.

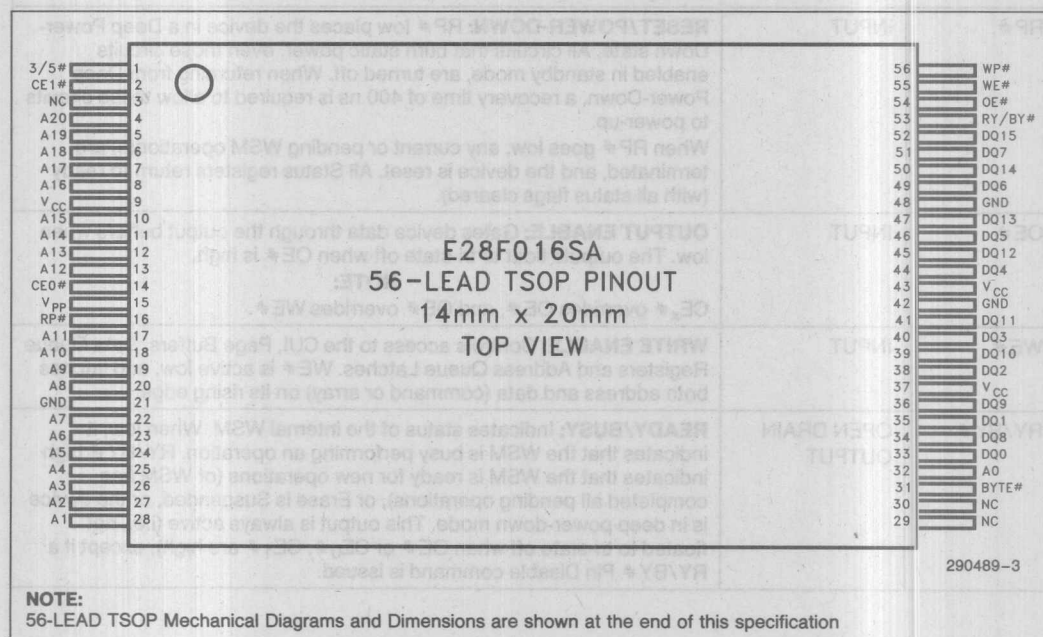


Figure 2. TSOP Pinout Configuration

Address	Mode	Block	Size	Block	Size
000000	RESERVED	000000	64 KByte Block	0	64 KByte Block
000000	RESERVED	000000	64 KByte Block	1	64 KByte Block
000000	RESERVED	000000	64 KByte Block	2	64 KByte Block
000000	RESERVED	000000	64 KByte Block	3	64 KByte Block
000000	RESERVED	000000	64 KByte Block	4	64 KByte Block
000000	RESERVED	000000	64 KByte Block	5	64 KByte Block
000000	RESERVED	000000	64 KByte Block	6	64 KByte Block
000000	RESERVED	000000	64 KByte Block	7	64 KByte Block
000000	RESERVED	000000	64 KByte Block	8	64 KByte Block
000000	RESERVED	000000	64 KByte Block	9	64 KByte Block
000000	RESERVED	000000	64 KByte Block	10	64 KByte Block
000000	RESERVED	000000	64 KByte Block	11	64 KByte Block
000000	RESERVED	000000	64 KByte Block	12	64 KByte Block
000000	RESERVED	000000	64 KByte Block	13	64 KByte Block
000000	RESERVED	000000	64 KByte Block	14	64 KByte Block
000000	RESERVED	000000	64 KByte Block	15	64 KByte Block
000000	RESERVED	000000	64 KByte Block	16	64 KByte Block
000000	RESERVED	000000	64 KByte Block	17	64 KByte Block
000000	RESERVED	000000	64 KByte Block	18	64 KByte Block
000000	RESERVED	000000	64 KByte Block	19	64 KByte Block
000000	RESERVED	000000	64 KByte Block	20	64 KByte Block
000000	RESERVED	000000	64 KByte Block	21	64 KByte Block
000000	RESERVED	000000	64 KByte Block	22	64 KByte Block
000000	RESERVED	000000	64 KByte Block	23	64 KByte Block
000000	RESERVED	000000	64 KByte Block	24	64 KByte Block
000000	RESERVED	000000	64 KByte Block	25	64 KByte Block
000000	RESERVED	000000	64 KByte Block	26	64 KByte Block
000000	RESERVED	000000	64 KByte Block	27	64 KByte Block
000000	RESERVED	000000	64 KByte Block	28	64 KByte Block
000000	RESERVED	000000	64 KByte Block	29	64 KByte Block
000000	RESERVED	000000	64 KByte Block	30	64 KByte Block
000000	RESERVED	000000	64 KByte Block	31	64 KByte Block

Figure 3. 28F016SA Memory Map (Byte-wide mode)

3.1 Extended Status Registers Memory Map

X8 MODE	A[20:0]	X16 MODE	A[20:1]
RESERVED	1F0006H	RESERVED	F8003H
GSR	1F0005H	GSR	F8002H
RESERVED	1F0004H	RESERVED	
BSR31	1F0003H	BSR31	F8001H
RESERVED	1F0002H	RESERVED	
RESERVED	1F0001H	RESERVED	F8000H
RESERVED	1F0000H	RESERVED	
•		•	
•		•	
•		•	
•		•	
•		•	
•		•	
•		•	
RESERVED	010002H	RESERVED	08001H
RESERVED	000006H	RESERVED	00003H
GSR	000005H	GSR	00002H
RESERVED	000004H	RESERVED	
BSR0	000003H	BSR0	00001H
RESERVED	000002H	RESERVED	
RESERVED	000001H	RESERVED	00000H
RESERVED	000000H	RESERVED	

Figure 4.1 Extended Status Register Memory Map (Byte-wide mode)

Figure 4.2 Extended Status Register Memory Map (Word-wide mode)

4.0 BUS OPERATIONS, COMMANDS AND STATUS REGISTER DEFINITIONS

4.1 Bus Operations for Word-Wide Mode (BYTE# = V_{IH})

Mode	Notes	RP#	CE ₁ #	CE ₀ #	OE#	WE#	A ₁	DQ ₀₋₁₅	RY/BY#
Read	1,2,7	V _{IH}	V _{IL}	V _{IL}	V _{IL}	V _{IH}	X	D _{OUT}	X
Output Disable	1,6,7	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{IH}	X	High Z	X
Standby	1,6,7	V _{IH}	V _{IL} V _{IH} V _{IH}	V _{IH} V _{IL} V _{IH}	X	X	X	High Z	X
Deep Power-Down	1,3	V _{IL}	X	X	X	X	X	High Z	V _{OH}
Manufacturer ID	4	V _{IH}	V _{IL}	V _{IL}	V _{IL}	V _{IH}	V _{IL}	0089H	V _{OH}
Device ID	4	V _{IH}	V _{IL}	V _{IL}	V _{IL}	V _{IH}	V _{IH}	66A0H	V _{OH}
Write	1,5,6	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{IL}	X	D _{IN}	X

4.2 Bus Operations For Byte-Wide Mode (BYTE# = V_{IL})

Mode	Notes	RP#	CE ₁ #	CE ₀ #	OE#	WE#	A ₀	DQ ₀₋₇	RY/BY#
Read	1,2,7	V _{IH}	V _{IL}	V _{IL}	V _{IL}	V _{IH}	X	D _{OUT}	X
Output Disable	1,6,7	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{IH}	X	High Z	X
Standby	1,6,7	V _{IH}	V _{IL} V _{IH} V _{IH}	V _{IH} V _{IL} V _{IH}	X	X	X	High Z	X
Deep Power-Down	1,3	V _{IL}	X	X	X	X	X	High Z	V _{OH}
Manufacturer ID	4	V _{IH}	V _{IL}	V _{IL}	V _{IL}	V _{IH}	V _{IL}	89H	V _{OH}
Device ID	4	V _{IH}	V _{IL}	V _{IL}	V _{IL}	V _{IH}	V _{IH}	A0H	V _{OH}
Write	1,5,6	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{IL}	X	D _{IN}	X

NOTES:

1. X can be V_{IH} or V_{IL} for address or control pins except for RY/BY#, which is either V_{OL} or V_{OH}.
2. RY/BY# output is open drain. When the WSM is ready, Erase is suspended or the device is in deep power-down mode, RY/BY# will be at V_{OH} if it is tied to V_{CC} through a resistor. When the RY/BY# at V_{OH} is independent of OE# while a WSM operation is in progress.
3. RP# at GND ± 0.2V ensures the lowest deep power-down current.
4. A₀ and A₁ at V_{IL} provide manufacturer ID codes in x8 and x16 modes respectively.
5. A₀ and A₁ at V_{IH} provide device ID codes in x8 and x16 modes respectively. All other addresses are set to zero.
6. Commands for different Erase operations, Data Write operations or Lock-Block operations can only be successfully completed when V_{PP} = V_{PPH}.
7. While the WSM is running, RY/BY# in Level-Mode (default) stays at V_{OL} until all operations are complete. RY/BY# goes to V_{OH} when the WSM is not busy or in erase suspend mode.
8. RY/BY# may be at V_{OL} while the WSM is busy performing various operations. For example, a status register read during a write operation.

4.3 28F008SA—Compatible Mode Command Bus Definitions

Command	Notes	First Bus Cycle			Second Bus Cycle		
		Oper	Addr	Data	Oper	Addr	Data
Read Array		Write	X	FFH	Read	AA	AD
intelligent identifier	1	Write	X	90H	Read	IA	ID
Read Compatible Status Register	2	Write	X	70H	Read	X	CSRD
Clear Status Register	3	Write	X	50H			
Word/Byte Write		Write	X	40H	Write	WA	WD
Alternate Word/Byte Write		Write	X	10H	Write	WA	WD
Block Erase/Confirm		Write	X	20H	Write	BA	D0H
Erase Suspend/Resume		Write	X	B0H	Write	X	D0H

ADDRESS

AA = Array Address
 BA = Block Address
 IA = Identifier Address
 WA = Write Address
 X = Don't Care

DATA

AD = Array Data
 CSRD = CSR Data
 ID = Identifier Data
 WD = Write Data

NOTES:

- Following the intelligent identifier command, two Read operations access the manufacturer and device signature codes.
- The CSR is automatically available after device enters Data Write, Erase, or Suspend operations.
- Clears CSR.3, CSR.4 and CSR.5. Also clears GSR.5 and all BSR.5 and BSR.2 bits. See Status register definitions.

4.4 28F016SA—Performance Enhancement Command Bus Definitions

Command	Mode	Notes	First Bus Cycle			Second Bus Cycle			Third Bus Cycle		
			Oper	Addr	Data	Oper	Addr	Data	Oper	Addr	Data
Read Extended Status Register		1	Write	X	71H	Read	RA	GSRD BSRD			
Page Buffer Swap		7	Write	X	72H						
Read Page Buffer			Write	X	75H	Read	PA	PD			
Single Load to Page Buffer			Write	X	74H	Write	PA	PD			
Sequential Load to Page Buffer	x8	4,6,10	Write	X	E0H	Write	X	BCL	Write	X	BCH
	x16	4,5,6,10	Write	X	E0H	Write	X	WCL	Write	X	WCH
Page Buffer Write to Flash	x8	3,4,9,10	Write	X	0CH	Write	A ₀	BC(L,H)	Write	WA	BC(H,L)
	x16	4,5,10	Write	X	0CH	Write	X	WCL	Write	WA	WCH
Two-Byte Write	x8	3	Write	X	FBH	Write	A ₀	WD(L,H)	Write	WA	WD(H,L)
Lock Block/Confirm			Write	X	77H	Write	BA	D0H			
Upload Status Bits/Confirm		2	Write	X	97H	Write	X	D0H			
Upload Device Information			Write	X	99H	Write	X	D0H			
Erase All Unlocked Blocks/Confirm			Write	X	A7H	Write	X	D0H			
RY/BY# Enable to Level-Mode		8	Write	X	96H	Write	X	01H			
RY/BY# Pulse-On-Write		8	Write	X	96H	Write	X	02H			
RY/BY# Pulse-On-Erase		8	Write	X	96H	Write	X	03H			
RY/BY# Disable		8	Write	X	96H	Write	X	04H			
Sleep			Write	X	F0H						
Abort			Write	X	80H						

ADDRESS

BA = Block Address
PA = Page Buffer Address
RA = Extended Register Address
WA = Write Address
X = Don't Care

DATA

AD = Array Data
PD = Page Buffer Data
BSRD = BSR Data
GSRD = GSR Data

WC (L,H) = Word Count (Low, High)
BC (L,H) = Byte Count (Low, High)
WD (L,H) = Write Data (Low, High)

NOTES:

1. RA can be the GSR address or any BSR address. See Figures 4.1 and 4.2 for Extended Status Register Memory Maps.
2. Upon device power-up, all BSR lock-bits come up locked. The Upload Status Bits command must be written to reflect the actual lock-bit status.
3. A₀ is automatically complemented to load second byte of data. BYTE # must be at V_{IL}.
4. A₀ value determines which WD/BC is supplied first: A₀ = 0 looks at the WDL/BCL, A₀ = 1 looks at the WDH/BCH.
5. BCH/WCH must be at 00H for this product because of the 256-Byte (128 Word) Page Buffer size and to avoid writing the Page Buffer contents into more than one 256-Byte segment within an array block. They are simply shown for future Page Buffer expandability.
6. In x16 mode, only the lower byte DQ₀₋₇ is used for WCL and WCH. The upper byte DQ₈₋₁₅ is a don't care.
7. PA and PD (whose count is given in cycles 2 and 3) are supplied starting in the 4th cycle which is not shown.
8. This command allows the user to swap between available Page Buffers (0 or 1).
9. These commands reconfigure RY/BY# output to one of two pulse-modes or enable and disable the RY/3Y# function.
10. Write address, WA, is the Destination address in the flash array which must match the Source address in the Page Buffer. Refer to the 28F016SA User's Manual.
11. BCL = 00H corresponds to a Byte count of 1. Similarly, WCL = 00H corresponds to a Word count of 1.

4.5 Compatible Status Register

WSMS	ESS	ES	DWS	VPPS	R	R	R
7	6	5	4	3	2	1	0

				NOTES:			
CSR.7 = WRITE STATE MACHINE STATUS				RY/BY# output or WSMS bit must be checked to determine completion of an operation (Erase Suspend, Erase or Data Write) before the appropriate Status bit (ESS, ES or DWS) is checked for success.			
1 = Ready							
0 = Busy							
CSR.6 = ERASE-SUSPEND STATUS							
1 = Erase Suspended							
0 = Erase in Progress/Completed							
CSR.5 = ERASE STATUS				If DWS and ES are set to "1" during an erase attempt, an improper command sequence was entered. Clear the CSR and attempt the operation again.			
1 = Error in Block Erasure							
0 = Successful Block Erase							
CSR.4 = DATA-WRITE STATUS							
1 = Error in Data Write							
0 = Data Write Successful							
CSR.3 = V _{PP} STATUS				The VPPS bit, unlike an A/D converter, does not provide continuous indication of V _{PP} level. The WSM interrogates V _{PP} 's level only after the Data-Write or Erase command sequences have been entered, and informs the system if V _{PP} has not been switched on. VPPS is not guaranteed to report accurate feedback between V _{PP} L and V _{PP} H.			
1 = V _{PP} Low Detect, Operation Abort							
0 = V _{PP} OK							

CSR.2-0 = RESERVED FOR FUTURE ENHANCEMENTS

These bits are reserved for future use; mask them out when polling the CSR.

4.6 Global Status Register

WSMS	OSS	DOS	DSS	QS	PBAS	PBS	PBSS
7	6	5	4	3	2	1	0

GSR.7 = WRITE STATE MACHINE STATUS

1 = Ready

0 = Busy

GSR.6 = OPERATION SUSPEND STATUS

1 = Operation Suspended

0 = Operation in Progress/Completed

GSR.5 = DEVICE OPERATION STATUS

1 = Operation Unsuccessful

0 = Operation Successful or Currently Running

GSR.4 = DEVICE SLEEP STATUS

1 = Device in Sleep

0 = Device Not in Sleep

MATRIX 5/4

00 = Operation Successful or Currently Running

01 = Device in Sleep Mode or Pending Sleep

10 = Operation Unsuccessful

11 = Operation Unsuccessful or Aborted

GSR.3 = QUEUE STATUS

1 = Queue Full

0 = Queue Available

GSR.2 = PAGE BUFFER AVAILABLE STATUS

1 = One or Two Page Buffers Available

0 = No Page Buffer Available

GSR.1 = PAGE BUFFER STATUS

1 = Selected Page Buffer Ready

0 = Selected Page Buffer Busy

GSR.0 = PAGE BUFFER SELECT STATUS

1 = Page Buffer 1 Selected

0 = Page Buffer 0 Selected

NOTES:

[1] RY/BY # output or WSMS bit must be checked to determine completion of an operation (Block Lock, Suspend, any RY/BY # reconfiguration, Upload Status Bits, Erase or Data Write) before the appropriate Status bit (OSS or DOS) is checked for success.

If operation currently running, then GSR.7 = 0.
If device pending sleep, then GSR.7 = 0.

Operation aborted: Unsuccessful due to Abort command.

The device contains two Page Buffers.

Selected Page Buffer is currently busy with WSM operation

NOTE:

1. When multiple operations are queued, checking BSR.7 only provides indication of completion for that particular block. GSR.7 provides indication when all queued operations are completed.

BS	BLS	BOS	BOAS	QS	VPPS	R	R
7	6	5	4	3	2	1	0

BSR.7 = BLOCK STATUS

- 1 = Ready
- 0 = Busy

BSR.6 = BLOCK-LOCK STATUS

- 1 = Block Unlocked for Write/Erase
- 0 = Block Locked for Write/Erase

BSR.5 = BLOCK OPERATION STATUS

- 1 = Operation Unsuccessful
- 0 = Operation Successful or Currently Running

BSR.4 = BLOCK OPERATION ABORT STATUS

- 1 = Operation Aborted
- 0 = Operation Not Aborted

MATRIX 5/4

- 00 = Operation Successful or Currently Running
- 01 = Not a Valid Combination
- 10 = Operation Unsuccessful
- 11 = Operation Aborted

BSR.3 = QUEUE STATUS

- 1 = Queue Full
- 0 = Queue Available

BSR.2 = V_{PP} STATUS

- 1 = V_{PP} Low Detect, Operation Abort
- 0 = V_{PP} OK

NOTES:

[1] RY/BY# output or BS bit must be checked to determine completion of an operation (Block Lock, Suspend, Erase or Data Write) before the appropriate Status bits (BOS, BLS) is checked for success.

The BOAS bit will not be set until BSR.7 = 1.

Operation halted via Abort command.

NOTES:

BSR.1-0 = RESERVED FOR FUTURE ENHANCEMENTS

These bits are reserved for future use; mask them out when polling the BSRs.

1. When multiple operations are queued, checking BSR.7 only provides indication of completion for that particular block. GSR.7 provides indication when all queued operations are completed.

5.1 Absolute Maximum Ratings*

Temperature Under Bias 0°C to +80°C

Storage Temperature -65°C to +125°C

products in the sampling and initial production phases of development. The specifications are subject to change without notice. Verify with your local Intel Sales office that you have the latest data sheet before finalizing a design.

***WARNING:** Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.

V_{CC} = 3.3V ± 0.3V Systems⁽⁵⁾

Symbol	Parameter	Notes	Min	Max	Units	Test Conditions
T _A	Operating Temperature, Commercial	1	0	70	°C	Ambient Temperature
V _{CC}	V _{CC} with Respect to GND	2	-0.2	7.0	V	
V _{PP}	V _{PP} Supply Voltage with Respect to GND	2,3	-0.2	14.0	V	
V	Voltage on any Pin (except V _{CC} , V _{PP}) with Respect to GND	2	-0.5	V _{CC} + 0.5	V	
I	Current into any Non-Supply Pin			± 30	mA	
I _{OUT}	Output Short Circuit Current	4		100	mA	

V_{CC} = 5.0V ± 0.5V, V_{CC} = 5.0V ± 0.25V Systems^(5,6)

Symbol	Parameter	Notes	Min	Max	Units	Test Conditions
T _A	Operating Temperature, Commercial	1	0	70	°C	Ambient Temperature
V _{CC}	V _{CC} with Respect to GND	2	-0.2	7.0	V	
V _{PP}	V _{PP} Supply Voltage with Respect to GND	2,3	-0.2	14.0	V	
V	Voltage on any Pin (except V _{CC} , V _{PP}) with Respect to GND	2	-2.0	7.0	V	
I	Current into any Non-Supply Pin			± 30	mA	
I _{OUT}	Output Short Circuit Current	4		100	mA	

NOTES:

- Operating temperature is for commercial product defined by this specification.
- Minimum DC voltage is -0.5V on input/output pins. During transitions, this level may undershoot to -2.0V for periods <20 ns. Maximum DC voltage on input/output pins is V_{CC} + 0.5V which, during transitions, may overshoot to V_{CC} + 2.0V for periods <20 ns.
- Maximum DC voltage on V_{PP} may overshoot to +14.0V for periods <20ns.
- Output shorted for no more than one second. No more than one output shorted at a time.
- AC specifications are valid at both voltage ranges. See DC Characteristics tables for voltage range-specific specifications.
- 5% V_{CC} specifications refer to the 28F016SA-70 in its High Speed Test configuration.

5.2 Capacitance

For a 3.3V System:

Symbol	Parameter	Note	Typ	Max	Units	Test Conditions
C _{IN}	Capacitance Looking into an Address/Control Pin	1	6	8	pF	T _A = 25°C, f = 1.0 MHz
C _{OUT}	Capacitance Looking into an Output Pin	1	8	12	pF	T _A = 25°C, f = 1.0 MHz
C _{LOAD}	Load Capacitance Driven by Outputs for Timing Specifications	1		50	pF	For V _{CC} = 3.3V ± 0.3V
	Equivalent Testing Load Circuit			2.5	ns	50Ω transmission line delay

For a 5.0V System:

Symbol	Parameter	Note	Typ	Max	Units	Test Conditions
C _{IN}	Capacitance Looking into an Address/Control Pin	1	6	8	pF	T _A = 25°C, f = 1.0 MHz
C _{OUT}	Capacitance Looking into an Output Pin	1	8	12	pF	T _A = 25°C, f = 1.0 MHz
C _{LOAD}	Load Capacitance Driven by Outputs for Timing Specifications	1		100	pF	For V _{CC} = 5.0V ± 0.5V
				30	pF	For V _{CC} = 5.0V ± 0.25V
	Equivalent Testing Load Circuit for V _{CC} ± 10%			2.5	ns	25Ω transmission line delay
	Equivalent Testing Load Circuit for V _{CC} ± 5%			2.5	ns	83Ω transmission line delay

NOTE:

1. Sampled, not 100% tested.

5.3 Timing Nomenclature

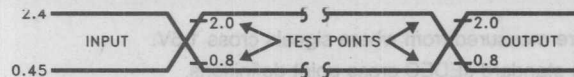
All 3.3V system timings are measured from where signals cross 1.5V.

For 5.0V systems use the standard JEDEC cross point definitions.

Each timing parameter consists of 5 characters. Some common examples are defined below:

t_{CE}	t_{ELQV} time(t) from CE # (E) going low (L) to the outputs (Q) becoming valid (V)
t_{OE}	t_{GLQV} time(t) from OE # (G) going low (L) to the outputs (Q) becoming valid (V)
t_{ACC}	t_{AVQV} time(t) from address (A) valid (V) to the outputs (Q) becoming valid (V)
t_{AS}	t_{AVWH} time(t) from address (A) valid (V) to WE # (W) going high (H)
t_{DH}	t_{WHDX} time(t) from WE # (W) going high (H) to when the data (D) can become undefined (X)

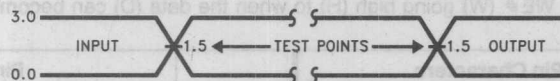
	Pin Characters		Pin States
A	Address Inputs	H	High
D	Data Inputs	L	Low
Q	Data Outputs	V	Valid
E	CE # (Chip Enable)	X	Driven, but not necessarily valid
G	OE # (Output Enable)	Z	High Impedance
W	WE # (Write Enable)		
P	RP # (Deep Power-Down Pin)		
R	RY/BY # (Ready Busy)		
V	Any Voltage Level		
Y	3/5 # Pin		
5V	V _{CC} at 4.5V Minimum		
3V	V _{CC} at 3.0V Minimum		



290489-4

AC test inputs are driven at V_{OH} (2.4 VTTL) for a Logic "1" and V_{OL} (0.45 VTTL) for a Logic "0". Input timing begins at V_{IH} (2.0 VTTL) and V_{IL} (0.8 VTTL). Output timing ends at V_{IH} and V_{IL} . Input rise and fall times (10% to 90%) < 10 ns.

Figure 5. Transient Input/Output Reference Waveform ($V_{CC} = 5.0V$) for Standard Test Configuration(1)



290489-5

AC test inputs are driven at 3.0V for a Logic "1" and 0.0V for a Logic "0". Input timing begins, and output timing ends, at 1.5V. Input rise and fall times (10% to 90%) < 10 ns.

Figure 6. Transient Input/Output Reference Waveform ($V_{CC} = 3.3V$) High Speed Reference Waveform(2) ($V_{CC} = 5.0V \pm 5\%$)

NOTES:

1. Testing characteristics for 28F016SA-080/28F016SA-100.
2. Testing characteristics for 28F016SA-070/28F016SA-120/28F016SA-150.

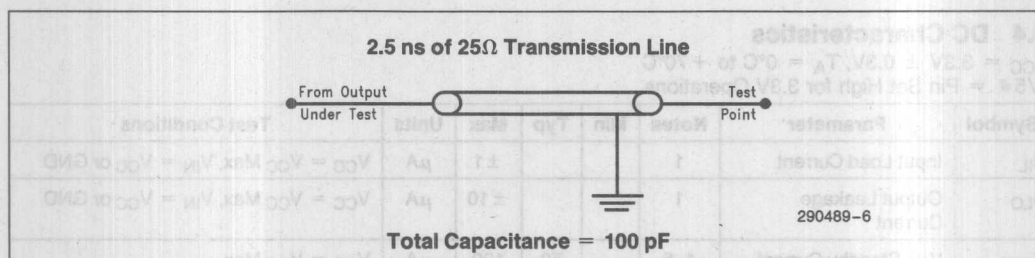


Figure 7. Transient Equivalent Testing Load Circuit ($V_{CC} = 5.0V$)

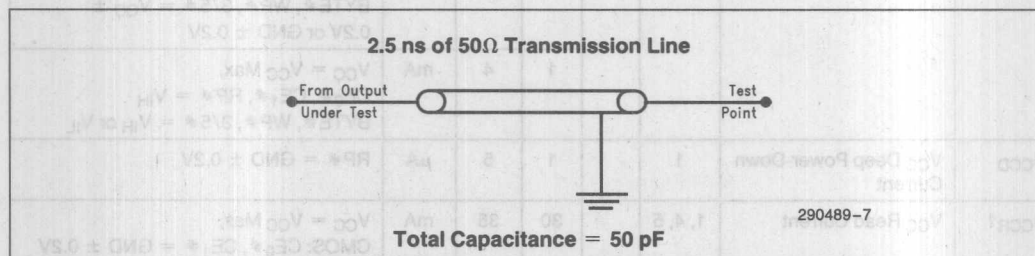


Figure 8. Transient Equivalent Testing Load Circuit ($V_{CC} = 3.3V$)

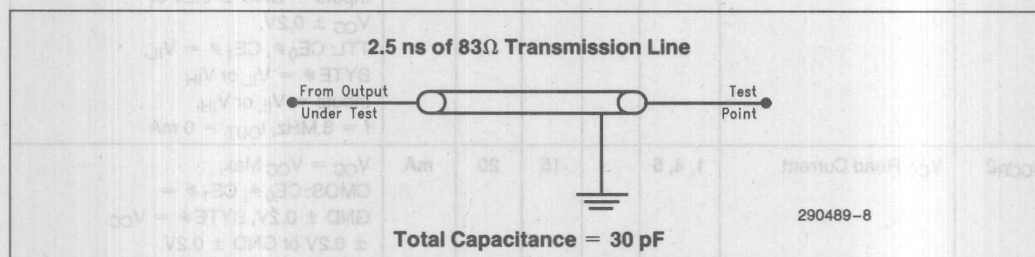


Figure 9. High Speed Transient Equivalent Testing Load Circuit ($V_{CC} = 5.0V \pm 5\%$)

3

5.4 DC Characteristics

 $V_{CC} = 3.3V \pm 0.3V$, $T_A = 0^\circ C$ to $+70^\circ C$

3/5# = Pin Set High for 3.3V Operations

Symbol	Parameter	Notes	Min	Typ	Max	Units	Test Conditions
I_{IL}	Input Load Current	1			± 1	μA	$V_{CC} = V_{CC} \text{ Max}$, $V_{IN} = V_{CC} \text{ or GND}$
I_{LO}	Output Leakage Current	1			± 10	μA	$V_{CC} = V_{CC} \text{ Max}$, $V_{IN} = V_{CC} \text{ or GND}$
I_{CCS}	V_{CC} Standby Current	1, 5		50	100	μA	$V_{CC} = V_{CC} \text{ Max}$, CE ₀ #, CE ₁ #, RP# = $V_{CC} \pm 0.2V$ BYTE#, WP#, 3/5# = $V_{CC} \pm 0.2V$ or GND $\pm 0.2V$
				1	4	mA	$V_{CC} = V_{CC} \text{ Max}$, CE ₀ #, CE ₁ #, RP# = V_{IH} BYTE#, WP#, 3/5# = V_{IH} or V_{IL}
I_{CCD}	V_{CC} Deep Power-Down Current	1		1	5	μA	RP# = GND $\pm 0.2V$
I_{CCR1}	V_{CC} Read Current	1, 4, 5		30	35	mA	$V_{CC} = V_{CC} \text{ Max}$, CMOS: CE ₀ #, CE ₁ # = GND $\pm 0.2V$ BYTE# = GND $\pm 0.2V$ or $V_{CC} \pm 0.2V$ Inputs = GND $\pm 0.2V$ or $V_{CC} \pm 0.2V$, TTL: CE ₀ #, CE ₁ # = V_{IL} , BYTE# = V_{IL} or V_{IH} Inputs = V_{IL} or V_{IH} , f = 8 MHz, $I_{OUT} = 0 \text{ mA}$
I_{CCR2}	V_{CC} Read Current	1, 4, 5		15	20	mA	$V_{CC} = V_{CC} \text{ Max}$, CMOS: CE ₀ #, CE ₁ # = GND $\pm 0.2V$, BYTE# = $V_{CC} \pm 0.2V$ or GND $\pm 0.2V$ Inputs = GND $\pm 0.2V$ or $V_{CC} \pm 0.2V$, TTL: CE ₀ #, CE ₁ # = V_{IL} , BYTE# = V_{IH} or V_{IL} Inputs = V_{IL} or V_{IH} , f = 4 MHz, $I_{OUT} = 0 \text{ mA}$
I_{CCW}	V_{CC} Write Current	1		8	12	mA	Word/Byte Write in Progress
I_{CCE}	V_{CC} Block Erase Current	1		6	12	mA	Block Erase in Progress
I_{CCES}	V_{CC} Erase Suspend Current	1, 2		3	6	mA	CE ₀ #, CE ₁ # = V_{IH} Block Erase Suspended
I_{PPS}	V_{PP} Standby Current	1		± 1	± 10	μA	$V_{PP} \leq V_{CC}$
I_{PPD}	V_{PP} Deep Power-Down Current	1		0.2	5	μA	RP# = GND $\pm 0.2V$

5.4 DC Characteristics (Continued)

$V_{CC} = 3.3V \pm 0.3V$, $T_A = 0^\circ C$ to $+70^\circ C$

3/5# = Pin Set High for 3.3V Operations

Symbol	Parameter	Notes	Min	Typ	Max	Units	Test Conditions
I_{PPR}	V_{PP} Read Current	1	65		200	μA	$V_{PP} > V_{CC}$
I_{PPW}	V_{PP} Write Current	1		10	15	mA	$V_{PP} = V_{PPH}$, Word/Byte Write in Progress
I_{PPE}	V_{PP} Erase Current	1		4	10	mA	$V_{PP} = V_{PPH}$, Block Erase in Progress
I_{PPES}	V_{PP} Erase Suspend Current	1	65		200	μA	$V_{PP} = V_{PPH}$, Block Erase Suspended
V_{IL}	Input Low Voltage		-0.3		0.8	V	
V_{IH}	Input High Voltage		2.0		$V_{CC} + 0.3$	V	
V_{OL}	Output Low Voltage				0.4	V	$V_{CC} = V_{CC}$ Min and $I_{OL} = 4$ mA
V_{OH1}	Output High Voltage		2.4			V	$I_{OH} = -2.0$ mA $V_{CC} = V_{CC}$ Min
V_{OH2}			$V_{CC} - 0.2$				$I_{OH} = -100$ μA $V_{CC} = V_{CC}$ Min
V_{PPL}	V_{PP} during Normal Operations	3	0.0		6.5	V	
V_{PPH}	V_{PP} during Write/Erase Operations		11.4	12.0	12.6	V	
V_{LKO}	V_{CC} Erase/Write Lock Voltage		2.0			V	

NOTES:

1. All currents are in RMS unless otherwise noted. Typical values at $V_{CC} = 3.3V$, $V_{PP} = 12.0V$, $T = 25^\circ C$. These currents are valid for all product versions (package and speeds).
2. I_{CCES} is specified with the device de-selected. If the device is read while in erase suspend mode, current draw is the sum of I_{CCES} and I_{CCR} .
3. Block Erases, Word/Byte Writes and Lock Block operations are inhibited when $V_{PP} = V_{PPL}$ and not guaranteed in the range between V_{PPH} and V_{PPL} .
4. Automatic Power Savings (APS) reduces I_{CCR} to less than 1 mA in static operation.
5. CMOS Inputs are either $V_{CC} \pm 0.2V$ or $GND \pm 0.2V$. TTL Inputs are either V_{IL} or V_{IH} .

I_{CCS}	V_{CC} Standby Current	1	1.1	1.2	1.5	mA	$V_{CC} = V_{CC}$
I_{CCR}	V_{CC} Read Current	1.5	2	10	10	mA	$V_{CC} = V_{CC}$, $V_{PP} = V_{PPH}$
I_{CCW}	V_{CC} Write Current	1	25	25	25	mA	$V_{CC} = V_{CC}$, $V_{PP} = V_{PPH}$
I_{CCE}	V_{CC} Block Erase Current	1	25	25	25	mA	$V_{CC} = V_{CC}$, $V_{PP} = V_{PPH}$
I_{CCES}	V_{CC} Block Erase Suspend Current	1	10	10	10	mA	$V_{CC} = V_{CC}$, $V_{PP} = V_{PPH}$
I_{CCP}	V_{CC} Program Current	1	25	25	25	mA	$V_{CC} = V_{CC}$, $V_{PP} = V_{PPH}$

$V_{CC} = 5.0V \pm 0.5V$, $T_A = 0^\circ C$ to $+70^\circ C$
 3/5# Pin Set Low for 5V Operations

Symbol	Parameter	Notes	Min	Typ	Max	Units	Test Conditions
I_{IL}	Input Load Current	1			± 1	μA	$V_{CC} = V_{CC} \text{ Max}$ $V_{IN} = V_{CC} \text{ or GND}$
I_{LO}	Output Leakage Current	1			± 10	μA	$V_{CC} = V_{CC} \text{ Max}$ $V_{IN} = V_{CC} \text{ or GND}$
I_{CCS}	V_{CC} Standby Current	1, 5		50	100	μA	$V_{CC} = V_{CC} \text{ Max}$ $CE_0\#, CE_1\#, RP\# = V_{CC} \pm 0.2V$ $BYTE\#, WP\#, 3/5\# = V_{CC} \pm 0.2V \text{ or GND} \pm 0.2V$
				2	4	mA	$V_{CC} = V_{CC} \text{ Max}$ $CE_0\#, CE_1\#, RP\# = V_{IH}$ $BYTE\#, WP\#, 3/5\# = V_{IH} \text{ or } V_{IL}$
I_{CCD}	V_{CC} Deep Power-Down Current	1		1	5	μA	$RP\# = GND \pm 0.2V$
I_{CCR1}	V_{CC} Read Current	1, 4, 5		50	60	mA	$V_{CC} = V_{CC} \text{ Max}$, CMOS: $CE_0\#, CE_1\# = GND \pm 0.2V$ $BYTE\# = GND \pm 0.2V \text{ or } V_{CC} \pm 0.2V$ Inputs = $GND \pm 0.2V \text{ or } V_{CC} \pm 0.2V$, TTL: $CE_0\#, CE_1\# = V_{IL}$, $BYTE\# = V_{IL} \text{ or } V_{IH}$ Inputs = $V_{IL} \text{ or } V_{IH}$, $f = 10 \text{ MHz}$, $I_{OUT} = 0 \text{ mA}$
I_{CCR2}	V_{CC} Read Current	1, 4, 5		30	35	mA	$V_{CC} = V_{CC} \text{ Max}$, CMOS: $CE_0\#, CE_1\# = GND \pm 0.2V$ $BYTE\# = V_{CC} \pm 0.2V \text{ or } GND \pm 0.2V$ Inputs = $GND \pm 0.2V \text{ or } V_{CC} \pm 0.2V$ TTL: $CE_0\#, CE_1\# = V_{IL}$, $BYTE\# = V_{IH} \text{ or } V_{IL}$ Inputs = $V_{IL} \text{ or } V_{IH}$, $f = 5 \text{ MHz}$, $I_{OUT} = 0 \text{ mA}$
I_{CCW}	V_{CC} Write Current	1		25	35	mA	Word/Byte in Progress
I_{CCE}	V_{CC} Block Erase Current	1		18	25	mA	Block Erase in Progress
I_{CCES}	V_{CC} Erase Suspend Current	1, 2		5	10	mA	$CE_0\#, CE_1\# = V_{IH}$ Block Erase Suspended
I_{PPS}	V_{PP} Standby Current	1		± 1	± 10	μA	$V_{PP} \leq V_{CC}$

5.5 DC Characteristics (Continued)

 $V_{CC} = 5.0V \pm 0.5V$, $T_A = 0^{\circ}C$ to $+70^{\circ}C$

3/5# Pin Set Low for 5V Operations

Symbol	Parameter	Notes	Min	Typ	Max	Units	Test Conditions
I_{PPD}	V_{PP} Deep Power-Down Current	1		0.2	5	μA	$RP\# = GND \pm 0.2V$
I_{PPR}	V_{PP} Read Current	1		65	200	μA	$V_{PP} > V_{CC}$
I_{PPW}	V_{PP} Write Current	1		7	12	mA	$V_{PP} = V_{PPH}$ Word/Byte Write in Progress
I_{PPE}	V_{PP} Block Erase Current	1		5	10	mA	$V_{PP} = V_{PPH}$ Block Erase in Progress
I_{PPES}	V_{PP} Erase Suspend Current	1		65	200	μA	$V_{PP} = V_{PPH}$ Block Erase Suspended
V_{IL}	Input Low Voltage		-0.5		0.8	V	
V_{IH}	Input High Voltage		2.0		$V_{CC} + 0.5$	V	
V_{OL}	Output Low Voltage				0.45	V	$V_{CC} = V_{CC\ Min}$ $I_{OL} = 5.8\ mA$
V_{OH1}	Output High Voltage		0.85 V_{CC}			V	$I_{OH} = -2.5\ mA$ $V_{CC} = V_{CC\ Min}$
V_{OH2}			$V_{CC} - 0.4$				$I_{OH} = -100\ \mu A$ $V_{CC} = V_{CC\ Min}$
V_{PPL}	V_{PP} during Normal Operations	3	0.0		6.5	V	
V_{PPH}	V_{PP} during Erase/Write Operations		11.4	12.0	12.6	V	
V_{LKO}	V_{CC} Erase/Write Lock Voltage		2.0			V	

NOTES:

1. All currents are in RMS unless otherwise noted. Typical values at $V_{CC} = 5.0V$, $V_{PP} = 12.0V$, $T = 25^{\circ}C$. These currents are valid for all product versions (package and speeds).
2. I_{CCES} is specified with the device de-selected. If the device is read while in erase suspend mode, current draw is the sum of I_{CCES} and I_{CCR} .
3. Block Erases, Word/Byte Writes and Lock Block operations are inhibited when $V_{PP} = V_{PPL}$ and not guaranteed in the range between V_{PPH} and V_{PPL} .
4. Automatic Power Saving (APS) reduces I_{CCR} to less than 2 mA in Static operation.
5. CMOS Inputs are either $V_{CC} \pm 0.2V$ or $GND \pm 0.2V$. TTL Inputs are either V_{IL} or V_{IH} .

5.6 AC Characteristics—Read Only Operations⁽¹⁾V_{CC} = 3.3V ± 0.3V T_A = 0° to +70°C

Versions (4)			28F016SA-120		28F016SA-150		Units
Symbol	Parameter	Notes	Min	Max	Min	Max	
t _{AVAV}	Read Cycle Time		120		150		ns
t _{AVEL}	Address Setup to CE # Going Low	3	0		0		ns
t _{AVGL}	Address Setup to OE # Going Low	3	0		0		ns
t _{AVQV}	Address to Output Delay			120		150	ns
t _{ELQV}	CE # to Output Delay	2		120		150	ns
t _{PHQV}	RP # High to Output Delay			620		750	ns
t _{GLQV}	OE # to Output Delay	2		45		50	ns
t _{ELQX}	CE # to Output in Low Z	3	0		0		ns
t _{EHQZ}	CE # to Output in High Z	3		50		55	ns
t _{GLQX}	OE # to Output in Low Z	3	0		0		ns
t _{GHQZ}	OE # to Output in High Z	3		30		40	ns
t _{OH}	Output Hold from Address, CE # or OE # Change, Whichever Occurs First	3	0		0		ns
t _{FLQV} t _{FHQV}	BYTE # to Output Delay	3		120		150	ns
t _{FLQZ}	BYTE # Low to Output in High Z	3		30		40	ns
t _{ELFL} t _{ELFH}	CE # Low to BYTE # High or Low	3		5		5	ns

Versions(4)		V _{CC} ± 5%	28F016SA-070(5)						Units
		V _{CC} ± 10%			28F016SA-080(6)		28F016SA-100(6)		
Symbol	Parameter	Notes	Min	Max	Min	Max	Min	Max	
t _{AVAV}	Read Cycle Time		70		80		100		ns
t _{AVEL}	Address Setup to CE # Going Low	3	0		0		0		ns
t _{AVGL}	Address Setup to OE # Going Low	3	0		0		0		ns
t _{AVQV}	Address to Output Delay			70		80		100	ns
t _{ELQV}	CE # to Output Delay	2		70		80		100	ns
t _{PHQV}	RP # to Output Delay			400		480		550	ns
t _{GLQV}	OE # to Output Delay	2		30		35		40	ns
t _{ELQX}	CE # to Output in Low Z	3	0		0		0		ns
t _{EHQZ}	CE # to Output in High Z	3		25		30		35	ns
t _{GLQX}	OE # to Output in Low Z	3	0		0		0		ns
t _{GHQZ}	OE # to Output in High Z	3		25		30		35	ns
t _{OH}	Output Hold from Address, CE # or OE # Change, Whichever Occurs First	3	0		0		0		ns
t _{FLQV} t _{FHQV}	BYTE # to Output Delay	3		70		80		100	ns
t _{FLQZ}	BYTE # Low to Output in High Z	3		25		30		30	ns
t _{ELFL} t _{ELFH}	CE # Low to BYTE # High or Low	3		5		5		5	ns

NOTES:

1. See AC Input/Output Reference Waveforms for timing measurements, Figures 5 and 6.
2. OE # may be delayed up to t_{ELQV}–t_{GLQV} after the falling edge of CE # without impact on t_{ELQV}.
3. Sampled, not 100% tested.
4. Device Speeds are defined as:
 - 70/80 ns at V_{CC} = 5.0V equivalent to
 - 120 ns at V_{CC} = 3.3V
 - 100 ns at V_{CC} = 5.0V equivalent to
 - 150 ns at V_{CC} = 3.3V
5. See AC Input/Output Reference Waveforms and AC Testing Load Circuits for High Speed Test Configuration.
6. See Standard AC Input/Output Reference Waveforms and AC Testing Load Circuit.

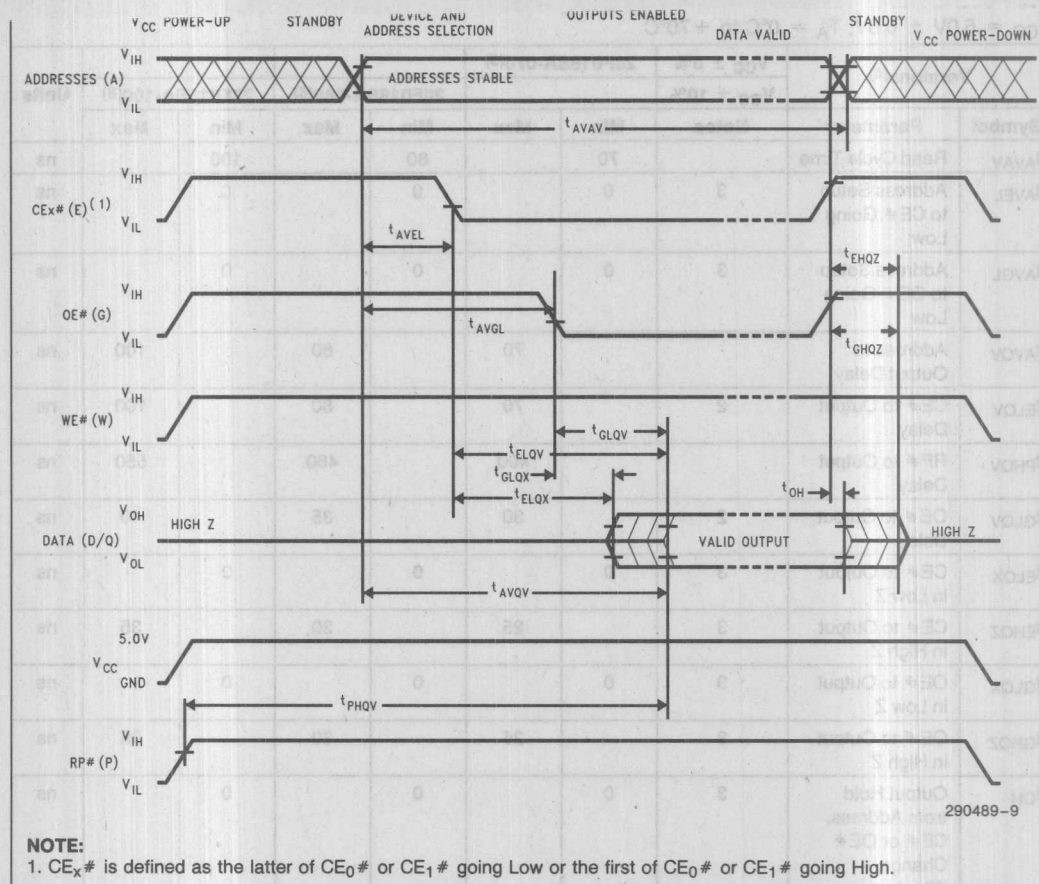


Figure 10. Read Timing Waveforms

100	60	70	3	Output High Z
30	30	25	3	Output High Z
5	5	5	3	Output High Z

NOTES:
1. See AC Input/Output Reference Waveforms for timing measurements, Figures 2 and 6.
2. CE_x may be driven up to 100 ns after the falling edge of CE₀ without impact on t_{AVAV}.
3. Samples not 100% tested.
4. Device speed and delay are defined as:
100 ns at V_{CC} = 2.0V equivalent to 150 ns at V_{CC} = 3.3V
120 ns at V_{CC} = 3.3V
150 ns at V_{CC} = 3.3V
5. See AC Input/Output Reference Waveforms and AC Testing Load Circuits for High Speed Test Configuration.
6. See Standard AC Input/Output Reference Waveforms and AC Testing Load Circuit.

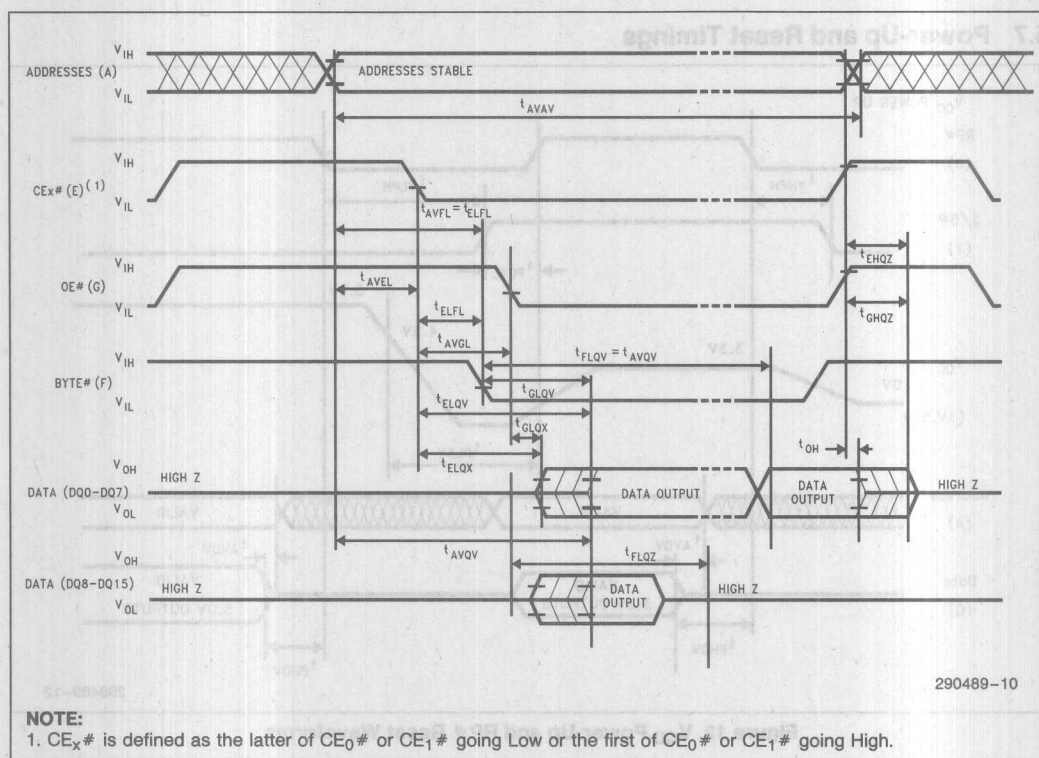


Figure 11. BYTE# Timing Waveforms

5.7 Power-Up and Reset Timings

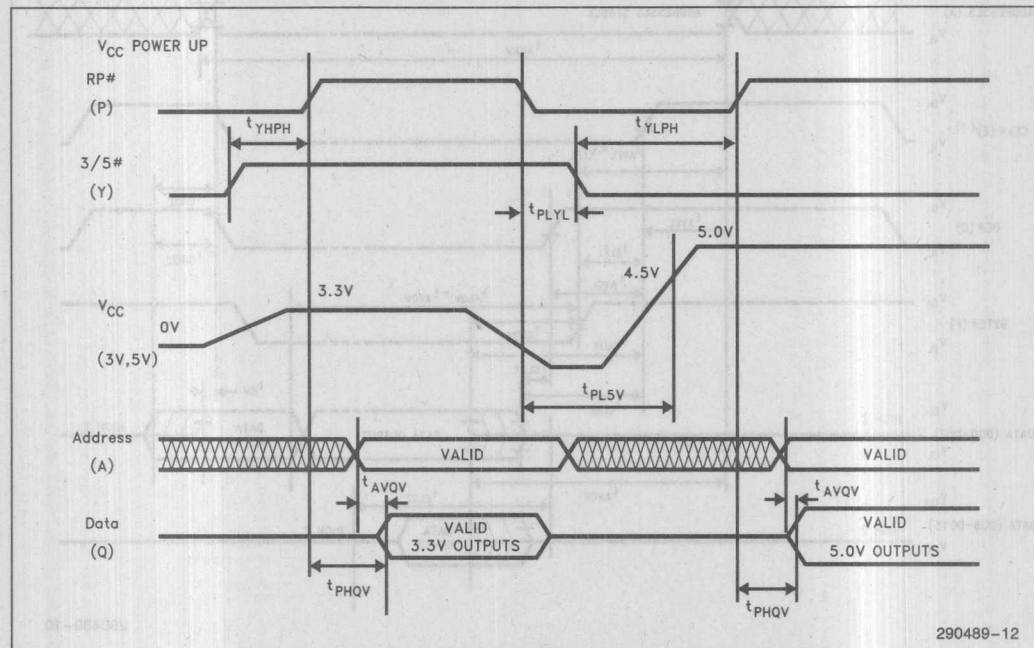


Figure 12. VCC Power-Up and RP# Reset Waveforms

Symbol	Parameter	Note	Min	Max	Unit
t_{PLYL} t_{PLYH}	RP# Low to 3/5# Low (High)		0		μs
t_{YLP} t_{YHP}	3/5# Low (High) to RP# High	1	2		μs
t_{PLSV} t_{PL3V}	RP# Low to VCC at 4.5V Minimum (to VCC at 3.0V min or 3.6V max)	2	0		μs
t_{AVQV}	Address Valid to Data Valid for VCC = 5V \pm 10%	3		80	ns
t_{PHQV}	RP# High to Data Valid for VCC = 5V \pm 10%	3		480	ns

NOTES:

CE₀#, CE₁# and OE# are switched low after Power-Up.

1. Minimum of 2 μs is required to meet the specified t_{PHQV} times.

2. The power supply may start to switch concurrently with RP# going Low.

3. The address access time and RP# high to data valid time are shown for 5V VCC operation. Refer to the AC Characteristics Read Only Operations 3.3V VCC operation and all other speed options.

$$V_{CC} = 3.3V \pm 0.3V \quad T_A = 0^\circ C \text{ to } +70^\circ C$$

Versions			28F016SA-120			28F016SA-150			Unit
Symbol	Parameter	Notes	Min	Typ	Max	Min	Typ	Max	
t _{AVAV}	Write Cycle Time		120			150			ns
t _{VPWH}	V _{pp} Setup to WE# Going High	3	100			100			ns
t _{PHEL}	RP# Setup to CE# Going Low		480			480			ns
t _{ELWL}	CE# Setup to WE# Going Low		10			10			ns
t _{AVWH}	Address Setup to WE# Going High	2, 6	75			75			ns
t _{DVWH}	Data Setup to WE# Going High	2, 6	75			75			ns
t _{WLWH}	WE# Pulse Width		75			75			ns
t _{WHDX}	Data Hold from WE# High	2	10			10			ns
t _{WHAX}	Address Hold from WE# High	2	10			10			ns
t _{WHEH}	CE# Hold from WE# High		10			10			ns
t _{WHWL}	WE# Pulse Width High		45			75			ns
t _{GHWL}	Read Recovery before Write		0			0			ns
t _{WHRL}	WE# High to RY/BY# Going Low				100			100	ns
t _{RHPL}	RP# Hold from Valid Status Register (CSR, GSR, BSR) Data and RY/BY# High	3	0			0			ns
t _{PHWL}	RP# High Recovery to WE# Going Low		1			1			μs
t _{WHGL}	Write Recovery before Read		95			120			ns
t _{QVVL}	V _{pp} Hold from Valid Status Register (CSR, GSR, BSR) Data and RY/BY# High		0			0			μs
t _{WHQV1}	Duration of Word/Byte Write Operation	4, 5	5	9		5	9		μs
t _{WHQV2}	Duration of Block Erase Operation	4	0.3			0.3			sec

3

Characteristics for WE# Controlled Command Write Operations⁽¹⁾ (Continued)
 $V_{CC} = 5.0V \pm 0.5V$ $T_A = 0^\circ C$ to $+70^\circ C$

Versions		V _{CC} ± 5%	28F016SA-070									Unit
		V _{CC} ± 10%				28F016SA-080			28F016SA-100			
Symbol	Parameter	Notes	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
t _{AVAV}	Write Cycle Time		70			80			100			ns
t _{VPWH}	V _{PP} Setup to WE # Going High	3	100			100			100			ns
t _{PHEL}	RP # Setup to CE # Going Low		480			480			480			ns
t _{ELWL}	CE # Setup to WE # Going Low		0			0			0			ns
t _{AVWH}	Address Setup to WE # Going High	2, 6	50			50			50			ns
t _{DVWH}	Data Setup to WE # Going High	2, 6	50			50			50			ns
t _{WLWH}	WE # Pulse Width		40			50			50			ns
t _{WHDX}	Data Hold from WE # High	2	0			0			0			ns
t _{WHAX}	Address Hold from WE # High	2	10			10			10			ns
t _{WHEH}	CE # Hold from WE # High		10			10			10			ns
t _{WHWL}	WE # Pulse Width High		30			30			50			ns
t _{GHWL}	Read Recovery before Write		0			0			0			ns
t _{WHRL}	WE # High to RY/BY # Going Low				100			100			100	ns
t _{RHPL}	RP # Hold from Valid Status Register (CSR, GSR, BSR) Data and RY/ BY # High	3	0			0			0			ns
t _{PHWL}	RP # High Recovery to WE # Going Low		1			1			1			μs
t _{WHGL}	Write Recovery before Read		60			65			80			ns
t _{QVVL}	V _{PP} Hold from Valid Status Register (CSR, GSR, BSR) Data and RY/BY # High		0			0			0			μs
t _{WHQV1}	Duration of Word/Byte Write Operation	4, 5	4.5	6		4.5	6		4.5	6		μs
t _{WHQV2}	Duration of Block Erase Operation	4	0.3			0.3			0.3			sec

NOTES:

CE# is defined as the latter of CE₀# or CE₁# going Low or the first of CE₀# or CE₁# going High.

1. Read timings during write and erase are the same as for normal read.
2. Refer to command definition tables for valid address and data values.
3. Sampled, but not 100% tested.
4. Write/Erase durations are measured to valid Status Register (CSR) Data.
5. Word/Byte write operations are typically performed with 1 Programming Pulse.
6. Address and Data are latched on the rising edge of WE# for all Command Write operations.

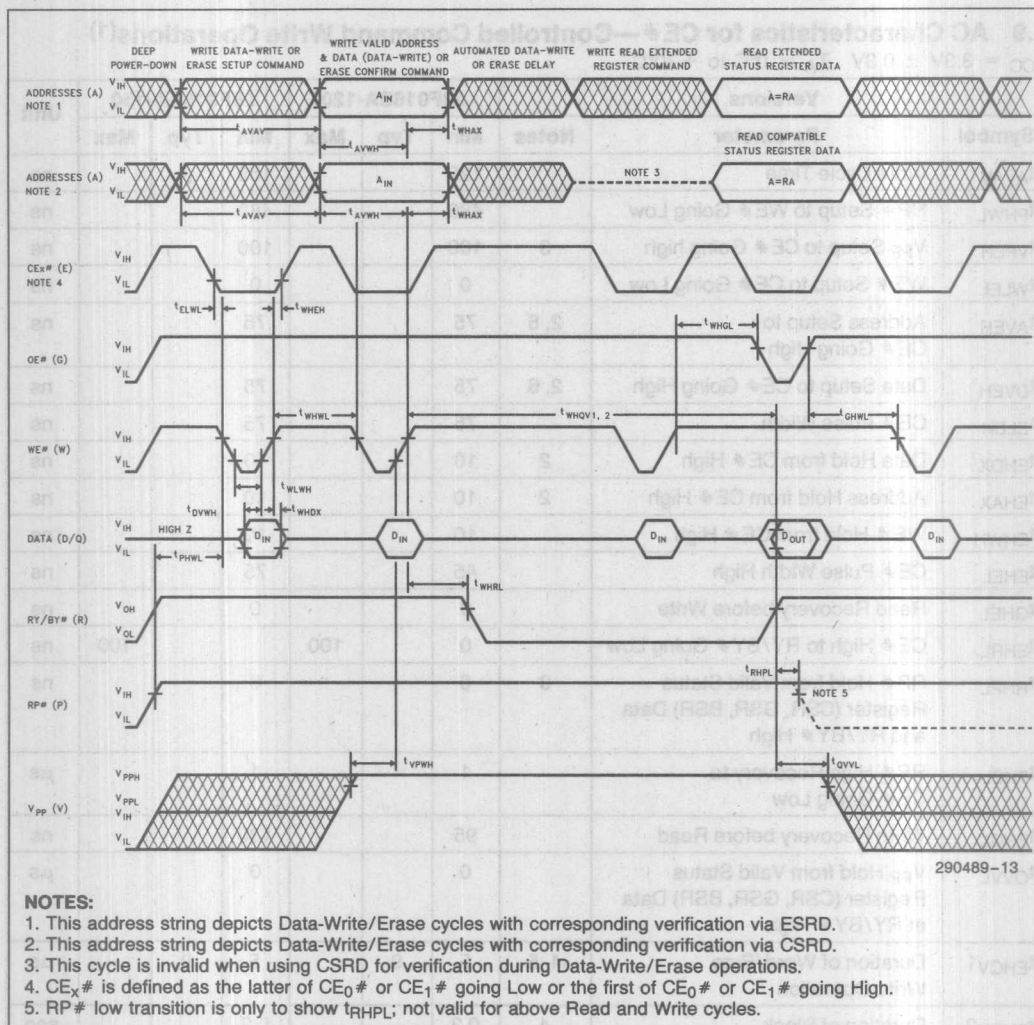


Figure 13. AC Waveforms for Command Write Operations

5.9 AC Characteristics for CE #—Controlled Command Write Operations⁽¹⁾ $V_{CC} = 3.3V \pm 0.3V$ $T_A = 0^{\circ}C$ to $+70^{\circ}C$

Versions			28F016SA-120			28F016SA-150			Unit
Symbol	Parameter	Notes	Min	Typ	Max	Min	Typ	Max	
t_{AVAV}	Write Cycle Time		120			150			ns
t_{PHWL}	RP # Setup to WE # Going Low		480			480			ns
t_{VPEH}	V_{PP} Setup to CE # Going high	3	100			100			ns
t_{WLEL}	WE # Setup to CE # Going Low		0			0			ns
t_{AVEH}	Address Setup to CE # Going High	2, 6	75			75			ns
t_{DVEH}	Data Setup to CE # Going High	2, 6	75			75			ns
t_{ELEH}	CE # Pulse Width		75			75			ns
t_{EHDx}	Data Hold from CE # High	2	10			10			ns
t_{EHAX}	Address Hold from CE # High	2	10			10			ns
t_{EHWL}	WE # Hold from CE # High		10			10			ns
t_{EHEL}	CE # Pulse Width High		45			75			ns
t_{GHLE}	Read Recovery before Write		0			0			ns
t_{EHRL}	CE # High to RY/BY # Going Low		0		100			100	ns
t_{RHPL}	RP # Hold from Valid Status Register (CSR, GSR, BSR) Data and RY/BY # High	3	0			0			ns
t_{PHLE}	RP # High Recovery to CE # Going Low		1			1			μs
t_{EHGL}	Write Recovery before Read		95			120			ns
t_{QVWL}	V_{PP} Hold from Valid Status Register (CSR, GSR, BSR) Data at RY/BY # High		0			0			μs
t_{EHQV1}	Duration of Word/Byte Write Operation	4, 5	5	9		5	9		μs
t_{EHQV2}	Duration of Block Erase Operation	4	0.3			0.3			sec

Versions		V _{CC} ± 5%	28F016SA-070									Unit
		V _{CC} ± 10%				28F016SA-080			28F016SA-100			
Symbol	Parameter	Notes	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
t _{AVAV}	Write Cycle Time		70			80			100			ns
t _{PHWL}	RP# Setup to WE# Going Low	3	100			480			480			ns
t _{VPEH}	V _{pp} Setup to CE# Going High	3	480			100			100			ns
t _{WLEL}	WE# Setup to CE# Going Low		0			0			0			ns
t _{AVEH}	Address Setup to CE# Going High	2, 6	50			50			50			ns
t _{DVEH}	Data Setup to CE# Going High	2, 6	50			50			50			ns
t _{ELEH}	CE# Pulse Width		40			50			50			ns
t _{EHDX}	Data Hold from CE# High	2	0			0			0			ns
t _{EHAX}	Address Hold from CE# High	2	10			10			10			ns
t _{EHWH}	WE# Hold from CE# High		10			10			10			ns
t _{EHCL}	CE# Pulse Width High		30			30			50			ns
t _{GHEL}	Read Recovery before Write		0			0			0			ns
t _{EHRL}	CE# High to RY/BY# Going Low				100			100			100	ns
t _{RHPL}	RP# Hold from Valid Status Register (CSR, GSR, BSR) Data and RY/BY# High	3	0			0			0			ns
t _{PHEL}	RP# High Recovery to CE# Going Low		1			1			1			μs
t _{EHGL}	Write Recovery before Read		60			65			80			ns
t _{QVVL}	V _{pp} Hold from Valid Status Register (CSR, GSR, BSR) Data and RY/BY# High		0			0			0			μs
t _{EHQV1}	Duration of Word/Byte Write Operation	4, 5	4.5	6		4.5	6		4.5	6		μs
t _{EHQV2}	Duration of Block Erase Operation	4	0.3			0.3			0.3			sec

NOTES:

CE# is defined as the latter of CE₀# or CE₁# going Low or the first of CE₀# or CE₁# going High.

1. Read timings during write and erase are the same as for normal read.
2. Refer to command definition tables for valid address and data values.
3. Sampled, but not 100% tested.
4. Write/Erase durations are measured to valid Status Data.
5. Word/Byte write operations are typically performed with 1 Programming Pulse.
6. Address and Data are latched on the rising edge of CE# for all Command Write Operations.

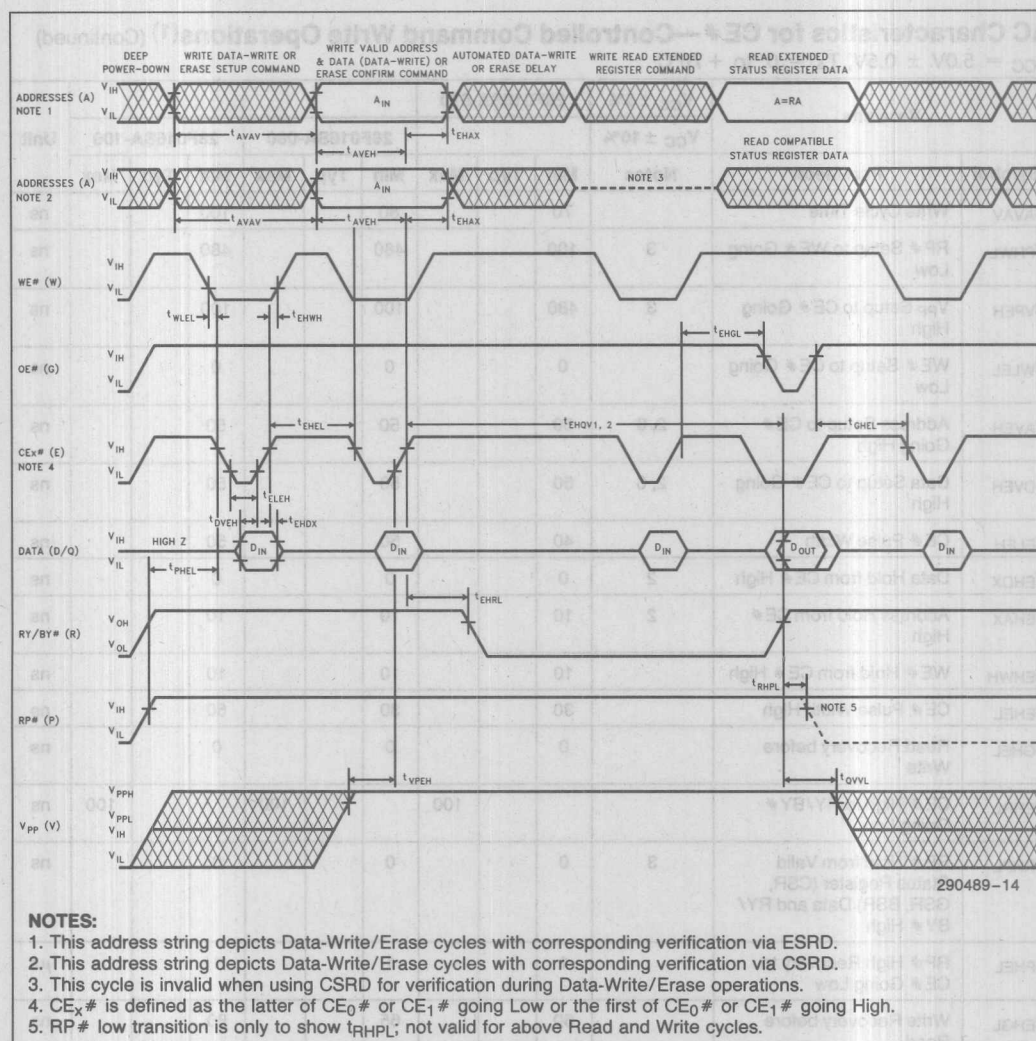


Figure 14. Alternate AC Waveforms for Command Write Operations

Symbol	Parameter	Value	Unit
t _{AVAV}	Address Valid to Address Valid	10	ns
t _{AVEH}	Address Valid to Address High	10	ns
t _{EHAX}	Address High to Address Valid	10	ns
t _{WLEL}	Write Enable Low to Write Enable High	10	ns
t _{EHWL}	Write Enable High to Write Enable Low	10	ns
t _{EHL}	Write Enable Low to Write Enable High	10	ns
t _{EHQV1,2}	Write Enable High to Write Enable Low	10	ns
t _{GHEL}	Write Enable Low to Write Enable High	10	ns
t _{PHEL}	Write Enable High to Write Enable Low	10	ns
t _{EHL}	Write Enable Low to Write Enable High	10	ns
t _{RHPL}	Write Enable High to Write Enable Low	10	ns
t _{VPEH}	Write Enable Low to Write Enable High	10	ns
t _{OVVL}	Write Enable High to Write Enable Low	10	ns

5.10 AC Characteristics for Page Buffer Write Operations⁽¹⁾

$V_{CC} = 3.3V \pm 0.3V$ $T_A = 0^\circ C$ to $+70^\circ C$.

Versions			28F016SA-120			28F016SA-150			Unit
Symbol	Parameter	Notes	Min	Typ	Max	Min	Typ	Max	
t_{AVAV}	Write Cycle Time		120			150			ns
t_{ELWL}	CE# Setup to WE# Going Low		10			10			ns
t_{AVWL}	Address Setup to WE# Going Low	3	0			0			ns
t_{DVWH}	Data Setup to WE# Going High	2	75			50			ns
t_{WLWH}	WE# Pulse Width		75			75			ns
t_{WHDX}	Data Hold from WE# High	2	10			10			ns
t_{WHAX}	Address Hold from WE# High	2	10			10			ns
t_{WHEH}	CE# Hold from WE# High		10			10			ns
t_{WHWL}	WE# Pulse Width High		45			75			ns
t_{GHWL}	Read Recovery before Write		0			0			ns
t_{WHGL}	Write Recovery before Read		95			120			ns

$V_{CC} = 5.0V \pm 0.5V$, $T_A = 0^\circ C$ to $+70^\circ C$

Versions			28F016SA-070			28F016SA-080			28F016SA-100			Unit
Symbol	Parameter	Notes	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
t_{AVAV}	Write Cycle Time		70			80			100			ns
t_{ELWL}	CE# Setup to WE# Going Low		0			0			0			ns
t_{AVWL}	Address Setup to WE# Going Low	3	0			0			0			ns
t_{DVWH}	Data Setup to WE# Going High	2	50			50			50			ns
t_{WLWH}	WE# Pulse Width		40			50			50			ns
t_{WHDX}	Data Hold from WE# High	2	0			0			0			ns
t_{WHAX}	Address Hold from WE# High	2	10			10			10			ns
t_{WHEH}	CE# Hold from WE# High		10			10			10			ns
t_{WHWL}	WE# Pulse Width High		30			30			50			ns
t_{GHWL}	Read Recovery before Write		0			0			0			ns
t_{WHGL}	Write Recovery before Read		60			65			80			ns

NOTES:

CE# is defined as the latter of CE₀# or CE₁# going Low or the first of CE₀# or CE₁# going High.

1. These are WE#-controlled write timings, equivalent CE#-controlled write timings apply.

2. Sampled, but not 100% tested.

3. Address must be valid during the entire WE# Low pulse.

5.1.1 Erase and word/Byte Write Performance⁽³⁾

$V_{CC} = 3.3V \pm 0.3V$ $T_A = 0^\circ C$ to $+70^\circ C$

Symbol	Parameter	Notes	Min	Typ ⁽¹⁾	Max	Units	Test Conditions
t_{WHRH1}	Word/Byte Write Time	2		9		μs	
t_{WHRH2}	Block Write Time	2		0.6	2.1	Sec	Byte Write Mode
t_{WHRH3}	Block Write Time	2		0.3	1.0	Sec	Word Write Mode
	Block Erase Time	2		0.8	10	Sec	
	Full Chip Erase Time	2		25.6		Sec	

$V_{CC} = 5.0V \pm 0.5V$ $T_A = 0^\circ C$ to $+70^\circ C$

Symbol	Parameter	Notes	Min	Typ ⁽¹⁾	Max	Units	Test Conditions
t_{WHRH1}	Word Byte/Write Time	2		6		μs	
t_{WHRH2}	Block Write Time	2		0.4	2.1	Sec	Byte Write Mode
t_{WHRH3}	Block Write Time	2		0.2	1.0	Sec	Word Write Mode
	Block Erase Time	2		0.6	10	Sec	
	Full Chip Erase Time	2		19.2		Sec	

NOTES:

1. $25^\circ C$, $V_{PP} = 12.0V$.
2. Excludes System-Level Overhead.
3. These performance numbers are valid for all speed versions.

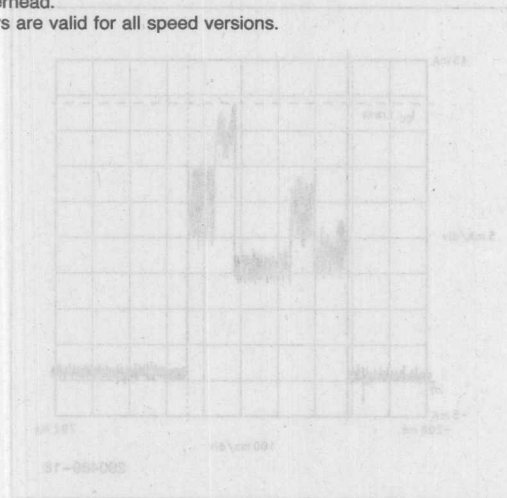


Figure 16: Icc during Block Erase

6.0 DERATING CURVES

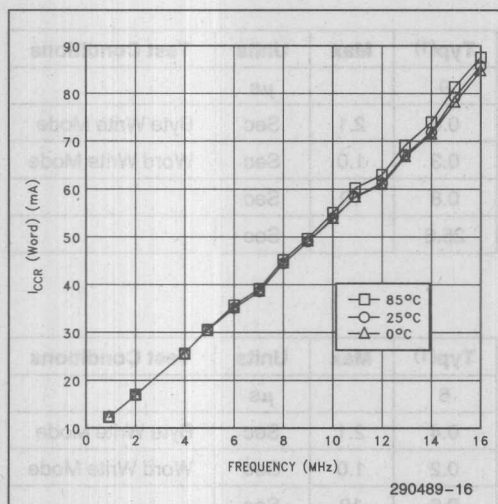


Figure 16. I_{CC} vs Frequency ($V_{CC} = 5.5V$)
for x8 or x16 Operation

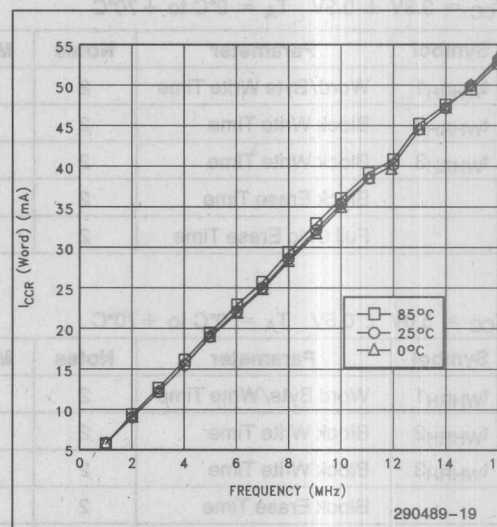


Figure 17. I_{CC} vs Frequency ($V_{CC} = 3.6V$)
for x8 or x16 Operation

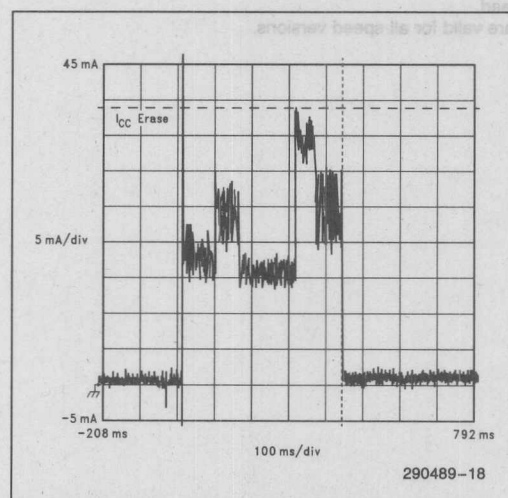


Figure 18. I_{CC} during Block Erase

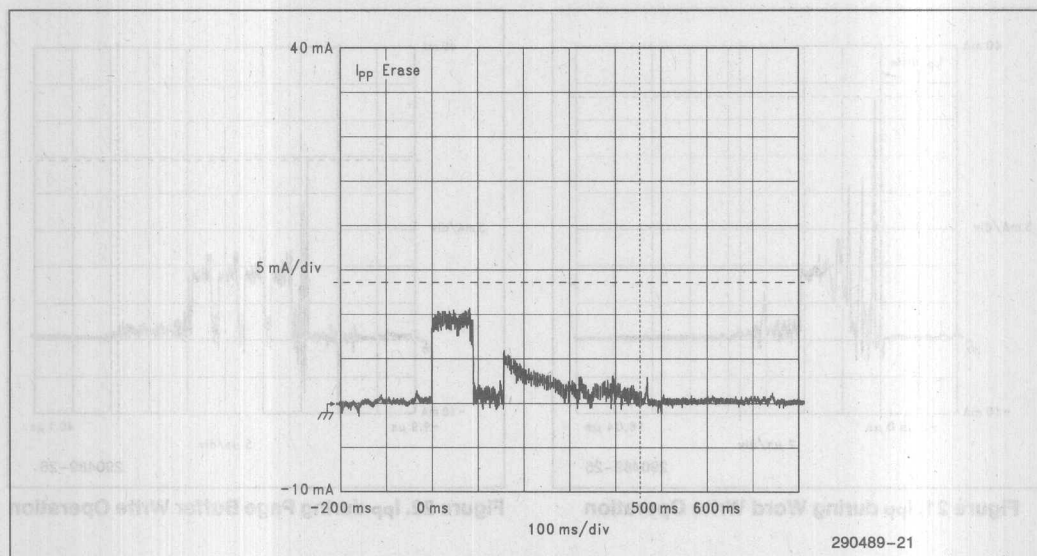


Figure 19. I_{pp} during Block Erase

3

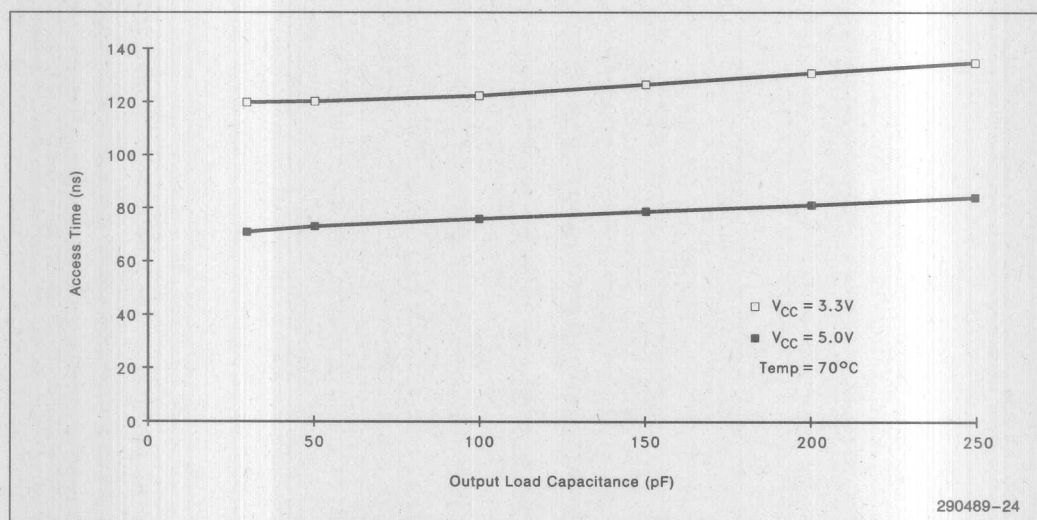


Figure 20. Access Time (t_{ACC}) vs Output Loading

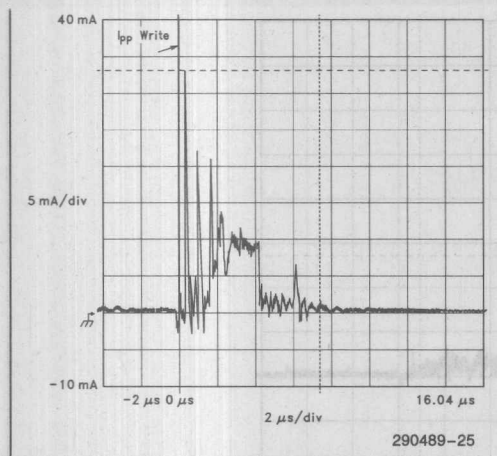


Figure 21. I_{pp} during Word Write Operation

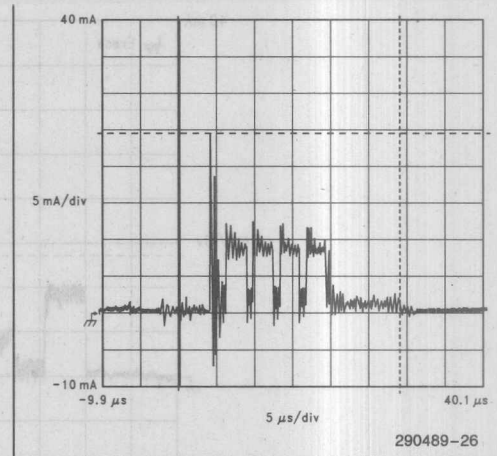


Figure 22. I_{pp} during Page Buffer Write Operation

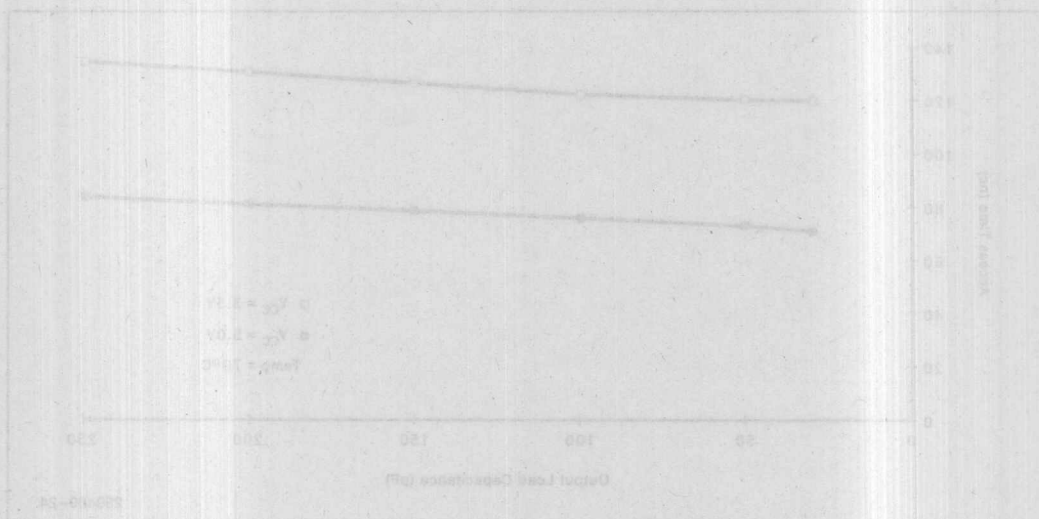


Figure 20. Access Time (t_{ACC}) vs Output Loading

7.0 MECHANICAL SPECIFICATIONS

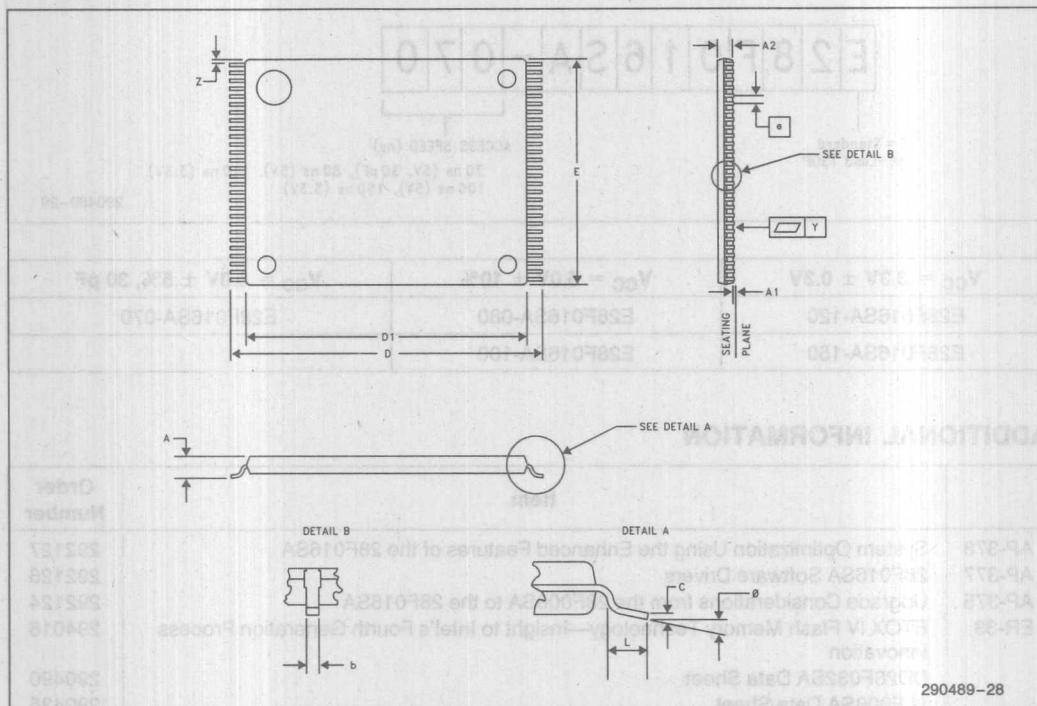


Figure 23. Mechanical Specifications of the 28F016SA 56-L-TSOP Type 1 Package

Family: Thin Small Outline Package				
Symbol	Millimeters			Notes
	Minimum	Nominal	Maximum	
A			1.20	
A ₁	0.50			
A ₂	0.965	0.995	1.025	
b	0.100	0.150	0.200	
c	0.115	0.125	0.135	
D ₁	18.20	18.40	18.60	
E	13.80	14.00	14.20	
e		0.50		
D	19.80	20.00	20.20	
L	0.500	0.600	0.700	
N		56		
Ø	0°	3°	5°	
Y			0.100	
Z	0.150	0.250	0.350	



28F008SA 8-MBIT (1-MBIT x 8) FLASHFILE™ MEMORY

- **High-Density Symmetrically Blocked Architecture**
 - Sixteen 64-Kbyte Blocks
- **Extended Cycling Capability**
 - 100,000 Block Erase Cycles
 - 1.6 Million Block Erase Cycles per Chip
- **Automated Byte Write and Block Erase**
 - Command User Interface
 - Status Register
- **System Performance Enhancements**
 - RY/BY# Status Output
 - Erase Suspend Capability
- **Deep-Powerdown Mode**
 - 0.20 μ A I_{CC} Typical
- **Very High-Performance Read**
 - 85 ns Maximum Access Time
- **SRAM-Compatible Write Interface**
- **Hardware Data Protection Feature**
 - Erase/Write Lockout during Power Transitions
- **Industry Standard Packaging**
 - 40-Lead TSOP, 44-Lead PSOP
- **ETOX III Nonvolatile Flash Technology**
 - 12V Byte Write/Block Erase
- **Independent Software Vendor Support**
 - Microsoft* Flash File System (FFS)

Intel's 28F008SA 8-Mbit FlashFile™ Memory is the highest density nonvolatile read/write solution for solid state storage. The 28F008SA's extended cycling, symmetrically blocked architecture, fast access time, write automation and low power consumption provide a more reliable, lower power, lighter weight and higher performance alternative to traditional rotating disk technology. The 28F008SA brings new capabilities to portable computing. Application and operating system software stored in resident flash memory arrays provide instant-on, rapid execute-in-place and protection from obsolescence through in-system software updates. Resident software also extends system battery life and increases reliability by reducing disk drive accesses.

For high density data acquisition applications, the 28F008SA offers a more cost-effective and reliable alternative to SRAM and battery. Traditional high density embedded applications, such as telecommunications, can take advantage of the 28F008SA's nonvolatility, blocking and minimal system code requirements for flexible firmware and modular software designs.

The 28F008SA is offered in 40-lead TSOP (standard and reverse) and 44-lead PSOP packages. Pin assignments simplify board layout when integrating multiple devices in a flash memory array or subsystem. This device uses an integrated Command User Interface and state machine for simplified block erasure and byte write. The 28F008SA memory map consists of 16 separately erasable 64-Kbyte blocks.

Intel's 28F008SA employs advanced CMOS circuitry for systems requiring low power consumption and noise immunity. Its 85 ns access time provides superior performance when compared with magnetic storage media. A deep powerdown mode lowers power consumption to 1 μ W typical thru V_{CC} , crucial in portable computing, handheld instrumentation and other low-power applications. The RP# power control input also provides absolute data protection during system powerup/down.

Manufactured on Intel's 0.8 micron ETOX process, the 28F008SA provides the highest levels of quality, reliability and cost-effectiveness.

*Microsoft is a trademark of Microsoft Corporation.

PRODUCT OVERVIEW

The 28F008SA is a high-performance **8-Mbit** (8,388,608 bit) memory organized as **1 Mbyte** (1,048,576 bytes) of 8 bits each. **Sixteen 64-Kbyte** (65,536 byte) **blocks** are included on the 28F008SA. A memory map is shown in Figure 6 of this specification. A block erase operation erases one of the sixteen blocks of memory in typically **1.6 seconds**, independent of the remaining blocks. Each block can be independently erased and written **100,000 cycles**. **Erase Suspend** mode allows system software to suspend block erase to read data or execute code from any other block of the 28F008SA.

The 28F008SA is available in the **40-lead TSOP** (Thin Small Outline Package, 1.2 mm thick) and **44-lead PSOP** (Plastic Small Outline) packages. Pin-outs are shown in Figures 2 and 4 of this specification.

The **Command User Interface** serves as the interface between the microprocessor or microcontroller and the internal operation of the 28F008SA.

Byte Write and Block Erase Automation allow byte write and block erase operations to be executed using a two-write command sequence to the Command User Interface. The internal **Write State Machine** (WSM) automatically executes the algorithms and timings necessary for byte write and block erase operations, including verifications, thereby unburdening the microprocessor or microcontroller. Writing of memory data is performed in byte increments typically within **9 μ s**, an 80% improvement over current flash memory products. **Ipp byte write and block erase currents** are **10 mA typical, 30 mA maximum**. **Vpp byte write and block erase voltage** is **11.4V to 12.6V**.

The **Status Register** indicates the status of the WSM and when the WSM successfully completes the desired byte write or block erase operation.

The **RY/BY#** output gives an additional indicator of WSM activity, providing capability for both hardware signal of status (versus software polling) and status masking (interrupt masking for background erase, for example). Status polling using RY/BY# minimizes both CPU overhead and system power consumption. When low, RY/BY# indicates that the WSM is performing a block erase or byte write operation. RY/BY# high indicates that the WSM is ready for new commands, block erase is suspended or the device is in deep powerdown mode.

Maximum access time is **85 ns (tACC)** over the commercial temperature range (0°C to +70°C) and over VCC supply voltage range (4.5V to 5.5V and 4.75V to 5.25V). **Icc active current** (CMOS Read) is **20 mA typical, 35 mA maximum at 8 MHz**.

When the CE# and RP# pins are at VCC, the **Icc CMOS Standby** mode is enabled.

A **Deep Powerdown** mode is enabled when the RP# pin is at GND, minimizing power consumption and providing write protection. **Icc current** in deep powerdown is **0.20 μ A typical**. Reset time of 400 ns is required from RP# switching high until outputs are valid to read attempts. Equivalently, the device has a wake time of 1 μ s from RP# high until writes to the Command User Interface are recognized by the 28F008SA. With RP# at GND, the WSM is reset and the Status Register is cleared.

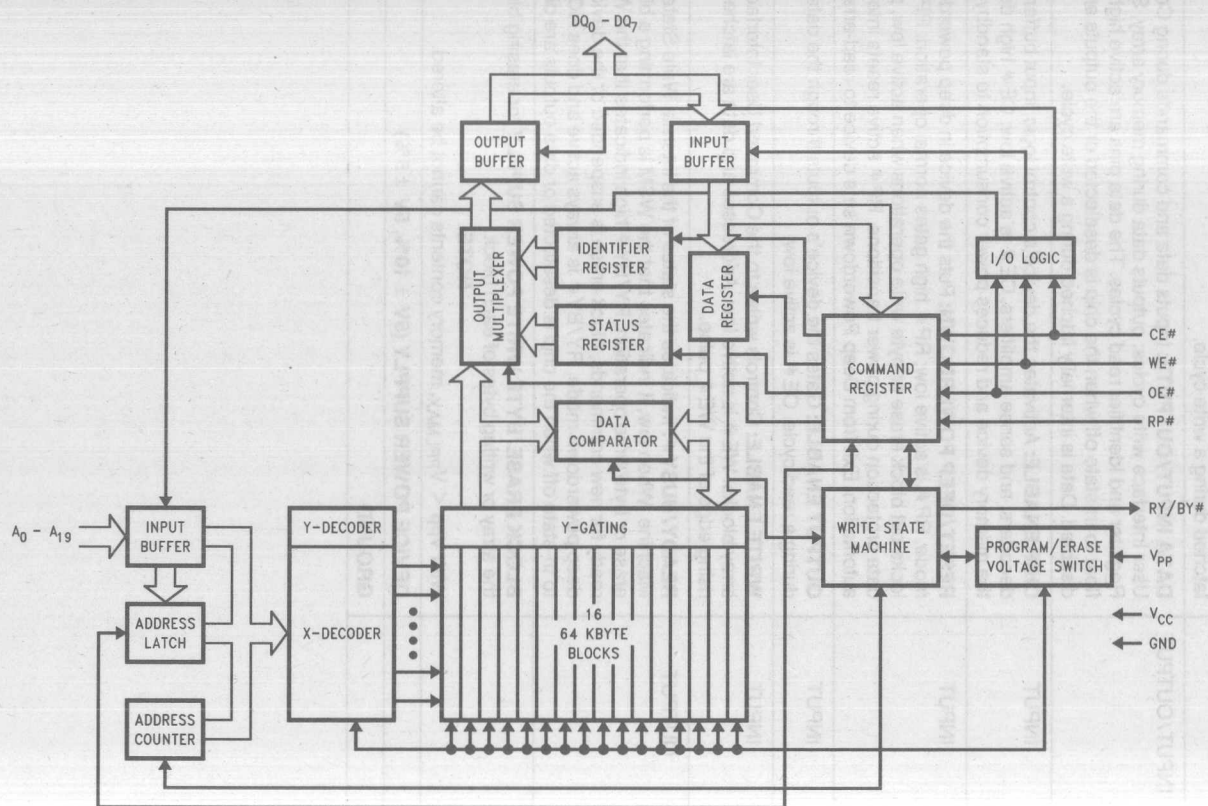
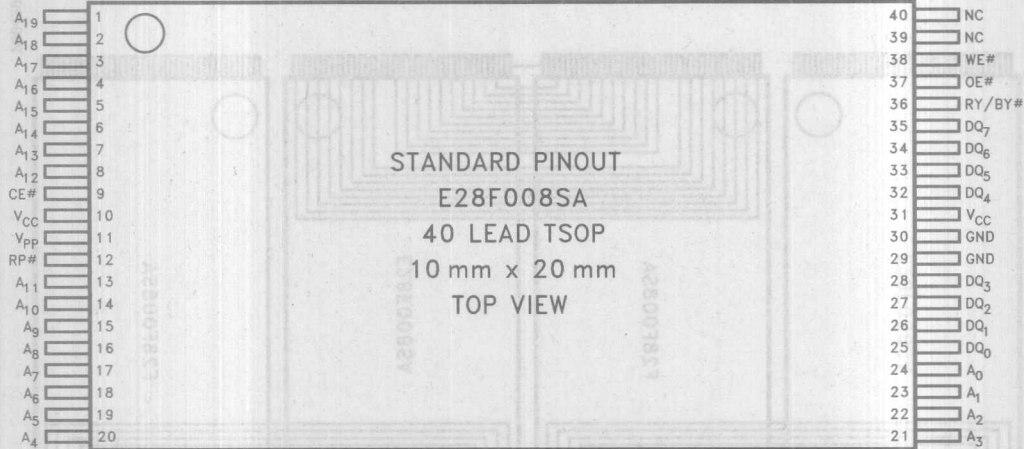


Figure 1. Block Diagram

Table 1. Pin Description

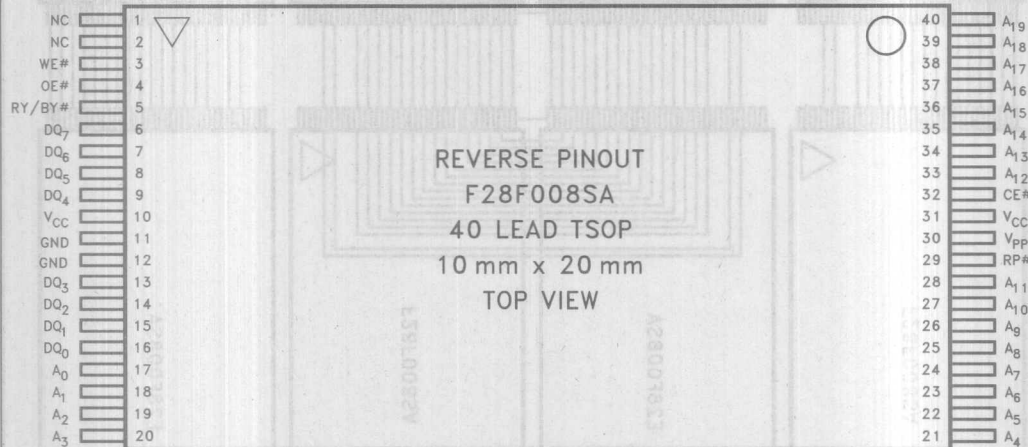
Symbol	Type	Name and Function
A ₀ –A ₁₉	INPUT	ADDRESS INPUTS for memory addresses. Addresses are internally latched during a write cycle.
DQ ₀ –DQ ₇	INPUT/OUTPUT	DATA INPUT/OUTPUTS: Inputs data and commands during Command User Interface write cycles; outputs data during memory array, Status Register and Identifier read cycles. The data pins are active high and float to tri-state off when the chip is deselected or the outputs are disabled. Data is internally latched during a write cycle.
CE #	INPUT	CHIP ENABLE: Activates the device's control logic, input buffers, decoders, and sense amplifiers. CE # is active low; CE # high deselects the memory device and reduces power consumption to standby levels.
RP #	INPUT	RESET/DEEP POWERDOWN: Puts the device in deep powerdown mode. RP # is active low; RP # high gates normal operation. RP # also locks out block erase or byte write operations when active low, providing data protection during power transitions. RP # active resets internal automation. Exit from Deep Powerdown sets device to read-array mode.
OE #	INPUT	OUTPUT ENABLE: Gates the device's outputs through the data buffers during a read cycle. OE # is active low.
WE #	INPUT	WRITE ENABLE: Controls writes to the Command User Interface and array blocks. WE # is active low. Addresses and data are latched on the rising edge of the WE # pulse.
RY/BY #	OUTPUT	READY/BUSY #: Indicates the status of the internal Write State Machine. When low, it indicates that the WSM is performing a block erase or byte write operation. RY/BY # high indicates that the WSM is ready for new commands, block erase is suspended or the device is in deep powerdown mode. RY/BY # is always active and does NOT float to tri-state off when the chip is deselected or data outputs are disabled.
V _{PP}		BLOCK ERASE/BYTE WRITE POWER SUPPLY for erasing blocks of the array or writing bytes of each block. NOTE: With V _{PP} < V _{PPLMAX} , memory contents cannot be altered.
V _{CC}		DEVICE POWER SUPPLY (5V ± 10%, 5V ± 5%)
GND		GROUND

Standard Pinout



290429-2

Reverse Pinout



290429-3

Figure 2. TSOP Lead Configurations

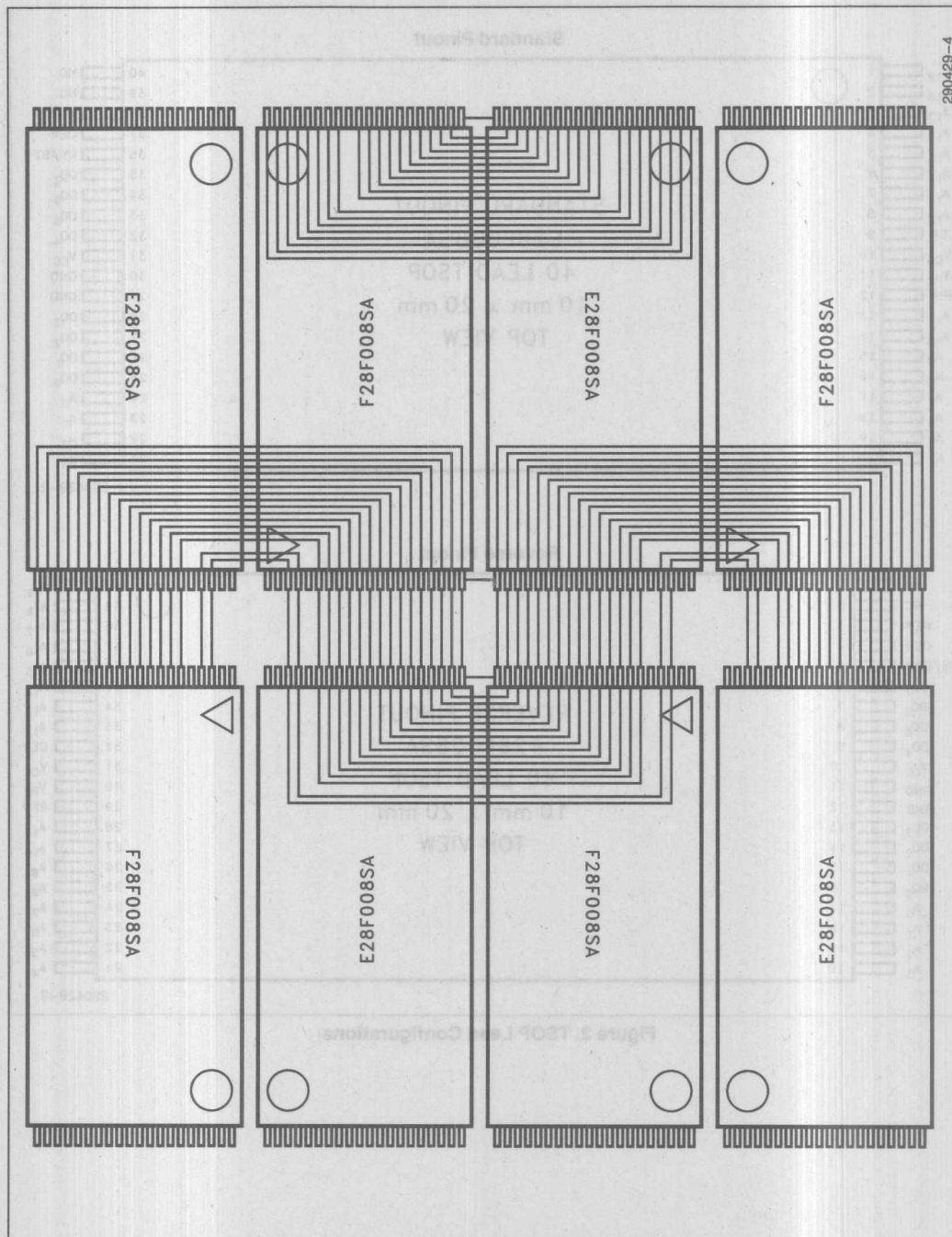


Figure 3. TSOP Serpentine Layout

NOTE:

1. Connect all V_{CC} and GND pins of each device to common power supply outputs. DO NOT leave V_{CC} or GND inputs disconnected.

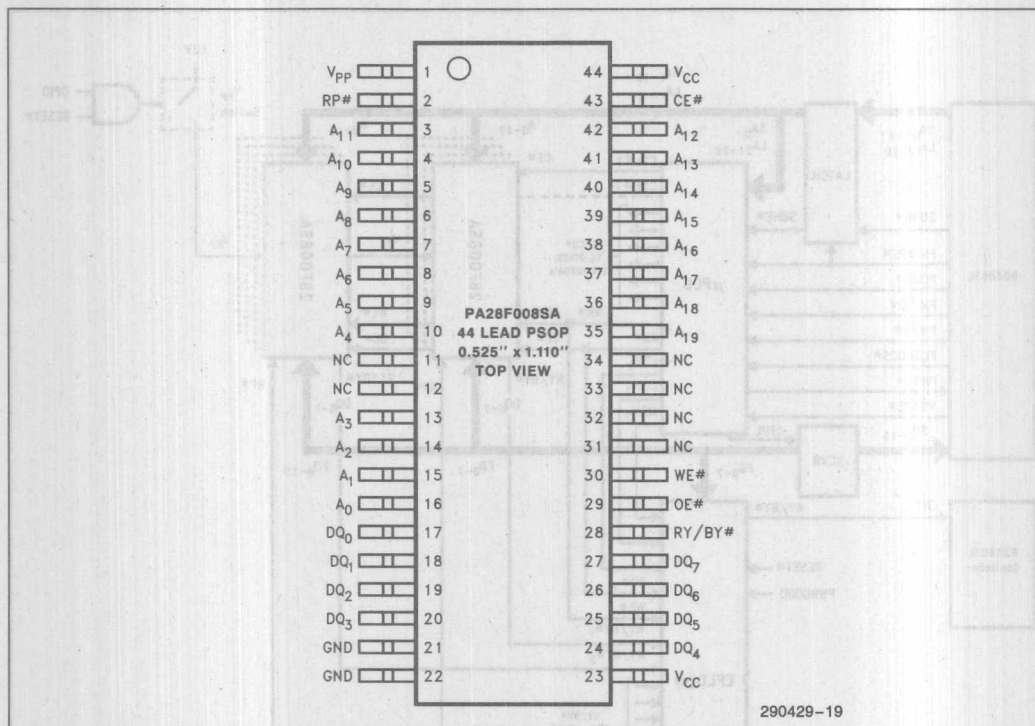


Figure 4. PSOP Lead Configuration

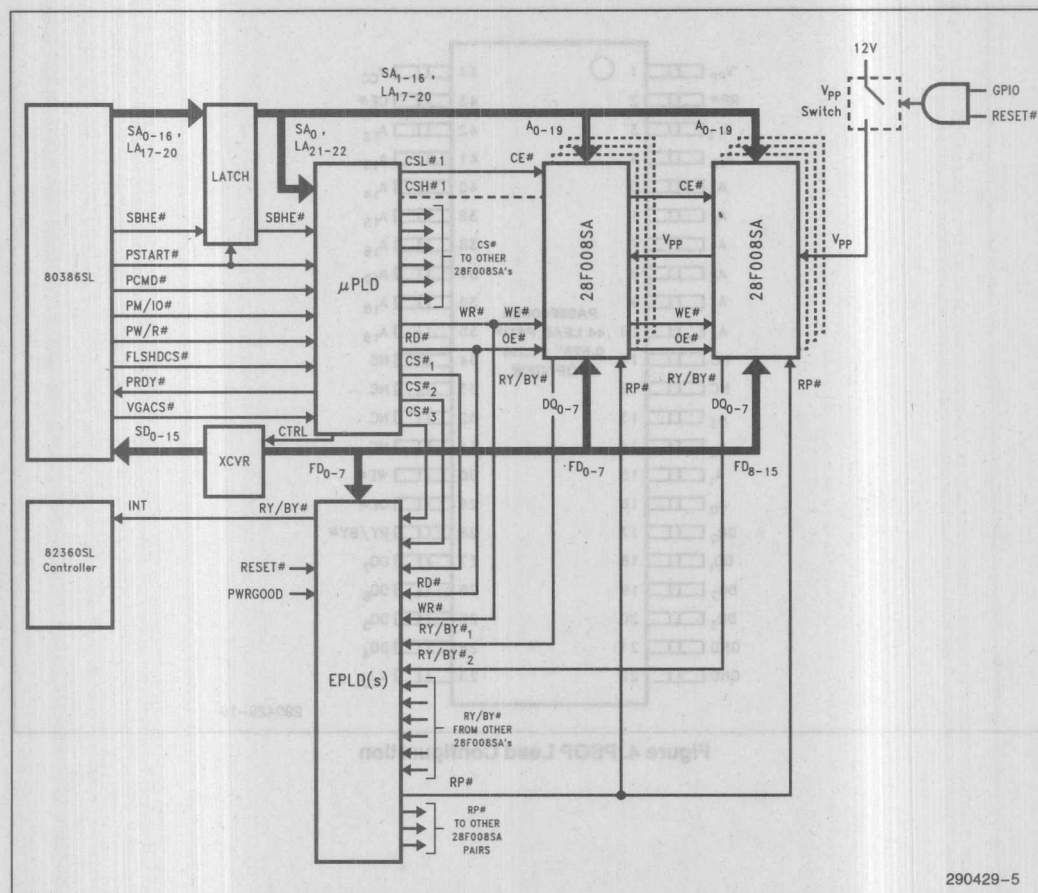


Figure 5. 28F008SA Array Interface to Intel386SL Microprocessor Superset through PI Bus (Including RY/BY # Masking and Selective Powerdown), for DRAM Backup during System SUSPEND, Resident O/S and Applications and Motherboard Solid-State Disk.

The 28F008SA includes on-chip write automation to manage write and erase functions. The Write State Machine allows for: 100% TTL-level control inputs; fixed power supplies during block erasure and byte write; and minimal processor overhead with RAM-like interface timings.

After initial device powerup, or after return from deep powerdown mode (see Bus Operations), the 28F008SA functions as a read-only memory. Manipulation of external memory-control pins allow array read, standby and output disable operations. Both Status Register and intelligent identifiers can also be accessed through the Command User Interface when $V_{PP} = V_{PPL}$.

This same subset of operations is also available when high voltage is applied to the V_{PP} pin. In addition, high voltage on V_{PP} enables successful block erasure and byte writing of the device. All functions associated with altering memory contents—byte write, block erase, status and intelligent identifier—are accessed via the Command User Interface and verified thru the Status Register.

Commands are written using standard microprocessor write timings. Command User Interface contents serve as input to the WSM, which controls the block erase and byte write circuitry. Write cycles also internally latch addresses and data needed for byte write or block erase operations. With the appropriate command written to the register, standard microprocessor read timings output array data, access the intelligent identifier codes, or output byte write and block erase status for verification.

Interface software to initiate and poll progress of internal byte write and block erase can be stored in any of the 28F008SA blocks. This code is copied to, and executed from, system RAM during actual flash memory update. After successful completion of byte write and/or block erase, code/data reads from the 28F008SA are again possible via the Read Array command. Erase suspend/resume capability allows system software to suspend block erase to read data and execute code from any other block.

FFFFF	64-Kbyte Block
F0000	
EFFFF	64-Kbyte Block
E0000	
DFFFF	64-Kbyte Block
D0000	
CFFFF	64-Kbyte Block
C0000	
BFFFF	64-Kbyte Block
B0000	
AFFFF	64-Kbyte Block
A0000	
9FFFF	64-Kbyte Block
90000	
8FFFF	64-Kbyte Block
80000	
7FFFF	64-Kbyte Block
70000	
6FFFF	64-Kbyte Block
60000	
5FFFF	64-Kbyte Block
50000	
4FFFF	64-Kbyte Block
40000	
3FFFF	64-Kbyte Block
30000	
2FFFF	64-Kbyte Block
20000	
1FFFF	64-Kbyte Block
10000	
0FFFF	64-Kbyte Block
00000	

Figure 6. Memory Map

Command User Interface and Write Automation

An on-chip state machine controls block erase and byte write, freeing the system processor for other tasks. After receiving the Erase Setup and Erase Confirm commands, the state machine controls block pre-conditioning and erase, returning progress via the Status Register and RY/BY# output. Byte write is similarly controlled, after destination address and expected data are supplied. The program and erase algorithms of past Intel flash memories are now regulated by the state machine, including pulse repetition where required and internal verification and margining of data.

Data Protection

Depending on the application, the system designer may choose to make the V_{PP} power supply switchable (available only when memory byte writes/block erases are required) or hardwired to V_{PPH} . When $V_{PP} = V_{PPL}$, memory contents cannot be altered. The 28F008SA Command User Interface architecture provides protection from unwanted byte write or block erase operations even when high voltage is applied to V_{PP} . Additionally, all functions are disabled whenever V_{CC} is below the write lockout voltage V_{LKO} , or when $RP\#$ is at V_{IL} . The 28F008SA accommodates either design practice and encourages optimization of the processor-memory interface.

The two-step byte write/block erase Command User Interface write sequence provides additional software write protection.

BUS OPERATION

Flash memory reads, erases and writes in-system via the local CPU. All bus cycles to or from the flash memory conform to standard microprocessor bus cycles.

Read

The 28F008SA has three read modes. The memory can be read from any of its blocks, and information can be read from the intelligent identifier or Status Register. V_{PP} can be at either V_{PPL} or V_{PPH} .

The first task is to write the appropriate read mode command to the Command User Interface (array, intelligent identifier, or Status Register). The 28F008SA automatically resets to Read Array mode upon initial device powerup or after exit from deep powerdown. The 28F008SA has four control pins, two of which must be logically active to obtain data at the outputs. Chip Enable ($CE\#$) is the device selection control, and when active enables the selected memory device. Output Enable ($OE\#$) is the data input/output (DQ_0 – DQ_7) direction control, and when active drives data from the selected memory onto the I/O bus. $RP\#$ and $WE\#$ must also be at V_{IH} . Figure 10 illustrates read bus cycle waveforms.

Output Disable

With $OE\#$ at a logic-high level (V_{IH}), the device outputs are disabled. Output pins (DQ_0 – DQ_7) are placed in a high-impedance state.

Standby

$CE\#$ at a logic-high level (V_{IH}) places the 28F008SA in standby mode. Standby operation disables much of the 28F008SA's circuitry and substantially reduces device power consumption. The outputs (DQ_0 – DQ_7) are placed in a high-impedance state independent of the status of $OE\#$. If the 28F008SA is deselected during block erase or byte write, the device will continue functioning and consuming normal active power until the operation completes.

Table 2. Bus Operations

Mode	Notes	$RP\#$	$CE\#$	$OE\#$	$WE\#$	A_0	V_{PP}	DQ_0 –7	$RY/BY\#$
Read	1, 2, 3	V_{IH}	V_{IL}	V_{IL}	V_{IH}	X	X	D_{OUT}	X
Output Disable	3	V_{IH}	V_{IL}	V_{IH}	V_{IH}	X	X	High Z	X
Standby	3	V_{IH}	V_{IH}	X	X	X	X	High Z	X
Deep PowerDown		V_{IL}	X	X	X	X	X	High Z	V_{OH}
Intelligent Identifier (Mfr)		V_{IH}	V_{IL}	V_{IL}	V_{IH}	V_{IL}	X	89H	V_{OH}
Intelligent Identifier (Device)		V_{IH}	V_{IL}	V_{IL}	V_{IH}	V_{IH}	X	A2H	V_{OH}
Write	3, 4, 5	V_{IH}	V_{IL}	V_{IH}	V_{IL}	X	X	D_{IN}	X

NOTES:

1. Refer to DC Characteristics. When $V_{PP} = V_{PPL}$, memory contents can be read but not written or erased.
2. X can be V_{IL} or V_{IH} for control pins and addresses, and V_{PPL} or V_{PPH} for V_{PP} . See DC Characteristics for V_{PPL} and V_{PPH} voltages.
3. $RY/BY\#$ is V_{OL} when the Write State Machine is executing internal block erase or byte write algorithms. It is V_{OH} when the WSM is not busy, in Erase Suspend mode or deep powerdown mode.
4. Command writes involving block erase or byte write are only successfully executed when $V_{PP} = V_{PPH}$.
5. Refer to Table 3 for valid D_{IN} during a write operation.

Deep Power-Down

The 28F008SA offers a deep power-down feature, entered when RP# is at V_{IL} . Current draw thru V_{CC} is 0.20 μA typical in deep power-down mode, with current draw through V_{pp} typically 0.1 μA . During read modes, RP#-low deselects the memory, places output drivers in a high-impedance state and turns off all internal circuits. The 28F008SA requires time t_{PHQV} (see AC Characteristics-Read-Only Operations) after return from powerdown until initial memory access outputs are valid. After this wakeup interval, normal operation is restored. The Command User Interface is reset to Read Array, and the upper 5 bits of the Status Register are cleared to value 10000, upon return to normal operation.

During block erase or byte write modes, RP# low will abort either operation. Memory contents of the block being altered are no longer valid as the data will be partially written or erased. Time t_{PHWL} after RP# goes to logic-high (V_{IH}) is required before another command can be written.

This use of RP# during system reset is important with automated write/erase devices. When the system comes out of reset it expects to read from the flash memory. Automated flash memories provide

status information when accessed during write/erase modes. If a CPU reset occurs with no flash memory reset, proper CPU initialization would not occur because the flash memory would be providing the status information instead of array data. Intel's Flash Memories allow proper CPU initialization following a system reset through the use of the RP# input. In this application RP# is controlled by the same RESET# signal that resets the system CPU.

Intelligent Identifier Operation

The intelligent identifier operation outputs the manufacturer code, 89H; and the device code, A2H for the 28F008SA. The system CPU can then automatically match the device with its proper block erase and byte write algorithms.

The manufacturer- and device-codes are read via the Command User Interface. Following a write of 90H to the Command User Interface, a read from address location 00000H outputs the manufacturer code (89H). A read from address 00001H outputs the device code (A2H). It is not necessary to have high voltage applied to V_{pp} to read the intelligent identifiers from the Command User Interface.

3

Table 3. Command Definitions

Command	Bus Cycles Req'd	Notes	First Bus Cycle			Second Bus Cycle		
			Operation	Address	Data	Operation	Address	Data
Read Array/Reset	1	1	Write	X	FFH			
Intelligent Identifier	3	2, 3, 4	Write	X	90H	Read	IA	IID
Read Status Register	2	3	Write	X	70H	Read	X	SRD
Clear Status Register	1		Write	X	50H			
Erase Setup/Erase Confirm	2	2	Write	BA	20H	Write	BA	D0H
Erase Suspend/Erase Resume	2		Write	X	B0H	Write	X	D0H
Byte Write Setup/Write	2	2, 3, 5	Write	WA	40H	Write	WA	WD
Alternate Byte Write Setup/Write	2	2, 3, 5	Write	WA	10H	Write	WA	WD

NOTES:

1. Bus operations are defined in Table 2.
2. IA = Identifier Address: 00H for manufacturer code, 01H for device code.
BA = Address within the block being erased.
WA = Address of memory location to be written.
3. SRD = Data read from Status Register. See Table 4 for a description of the Status Register bits.
WD = Data to be written at location WA. Data is latched on the rising edge of WE#.
IID = Data read from Intelligent Identifiers.
4. Following the Intelligent Identifier command, two read operations access manufacture and device codes.
5. Either 40H or 10H are recognized by the WSM as the Byte Write Setup command.
6. Commands other than those shown above are reserved by Intel for future device implementations and should not be used.

Write

Writes to the Command User Interface enable reading of device data and Intelligent Identifiers. They also control inspection and clearing of the Status Register. Additionally, when $V_{PP} = V_{PPH}$, the Command User Interface controls block erasure and byte write. The contents of the interface register serve as input to the internal state machine.

The Command User Interface itself does not occupy an addressable memory location. The interface register is a latch used to store the command and address and data information needed to execute the command. Erase Setup and Erase Confirm commands require both appropriate command data and an address within the block to be erased. The Byte Write Setup command requires both appropriate command data and the address of the location to be written, while the Byte Write command consists of the data to be written and the address of the location to be written.

The Command User Interface is written by bringing $WE\#$ to a logic-low level (V_{IL}) while $CE\#$ is low. Addresses and data are latched on the rising edge of $WE\#$. Standard microprocessor write timings are used.

Refer to AC Write Characteristics and the AC Waveforms for Write Operations, Figure 11, for specific timing parameters.

COMMAND DEFINITIONS

When V_{PPL} is applied to the V_{PP} pin, read operations from the Status Register, intelligent identifiers, or array blocks are enabled. Placing V_{PPH} on V_{PP} enables successful byte write and block erase operations as well.

Device operations are selected by writing specific commands into the Command User Interface. Table 3 defines the 28F008SA commands.

Read Array Command

Upon initial device powerup and after exit from deep powerdown mode, the 28F008SA defaults to Read Array mode. This operation is also initiated by writing FFH into the Command User Interface. Microprocessor read cycles retrieve array data. The device remains enabled for reads until the Command User Interface contents are altered. Once the internal Write State Machine has started a block erase or byte write operation, the device will not recognize the Read Array command, until the WSM has completed its operation. The Read Array command is functional when $V_{PP} = V_{PPL}$ or V_{PPH} .

Table 4. Status Register Definitions

	WSMS	ESS	ES	BWS	VPPS	R	R	R
	7	6	5	4	3	2	1	0
SR.7 = WRITE STATE MACHINE STATUS								
1 = Ready								
0 = Busy								
SR.6 = ERASE SUSPEND STATUS								
1 = Erase Suspended								
0 = Erase in Progress/Completed								
SR.5 = ERASE STATUS								
1 = Error in Block Erasure								
0 = Successful Block Erase								
SR.4 = BYTE WRITE STATUS								
1 = Error in Byte Write								
0 = Successful Byte Write								
SR.3 = V_{PP} STATUS								
1 = V_{PP} Low Detect; Operation Abort								
0 = V_{PP} OK								
SR.2-SR.0 = RESERVED FOR FUTURE ENHANCEMENTS								
These bits are reserved for future use and should be masked out when polling the Status Register.								
NOTES:								
RY/BY# or the Write State Machine Status bit must first be checked to determine byte write or block erase completion, before the Byte Write or Erase Status bit are checked for success.								
If the Byte Write AND Erase Status bits are set to "1"s during a block erase attempt, an improper command sequence was entered. Attempt the operation again.								
If V_{PP} low status is detected, the Status Register must be cleared before another byte write or block erase operation is attempted.								
The V_{PP} Status bit, unlike an A/D converter, does not provide continuous indication of V_{PP} level. The WSM interrogates the V_{PP} level only after the byte write or block erase command sequences have been entered and informs the system if V_{PP} has not been switched on. The V_{PP} Status bit is not guaranteed to report accurate feedback between V_{PPL} and V_{PPH} .								

Intelligent Identifier Command

The 28F008SA contains an Intelligent Identifier operation, initiated by writing 90H into the Command User Interface. Following the command write, a read cycle from address 00000H retrieves the manufacturer code of 89H. A read cycle from address 00001H returns the device code of A2H. To terminate the operation, it is necessary to write another valid command into the register. Like the Read Array command, the Intelligent Identifier command is functional when $V_{pp} = V_{ppl}$ or V_{pph} .

Read Status Register Command

The 28F008SA contains a Status Register which may be read to determine when a byte write or block erase operation is complete, and whether that operation completed successfully. The Status Register may be read at any time by writing the Read Status Register command (70H) to the Command User Interface. After writing this command, all subsequent read operations output data from the Status Register, until another valid command is written to the Command User Interface. The contents of the Status Register are latched on the falling edge of OE# or CE#, whichever occurs last in the read cycle. OE# or CE# must be toggled to V_{IH} before further reads to update the Status Register latch. The Read Status Register command functions when $V_{pp} = V_{ppl}$ or V_{pph} .

Clear Status Register Command

The Erase Status and Byte Write Status bits are set to "1"s by the Write State Machine and can only be reset by the Clear Status Register Command. These bits indicate various failure conditions (see Table 4). By allowing system software to control the resetting of these bits, several operations may be performed (such as cumulatively writing several bytes or erasing multiple blocks in sequence). The Status Register may then be polled to determine if an error occurred during that sequence. This adds flexibility to the way the device may be used.

Additionally, the V_{pp} Status bit (SR.3) MUST be reset by system software before further byte writes or block erases are attempted. To clear the Status Register, the Clear Status Register command (50H) is written to the Command User Interface. The Clear Status Register command is functional when $V_{pp} = V_{ppl}$ or V_{pph} .

Erase Setup/Erase Confirm Commands

Erase is executed one block at a time, initiated by a two-cycle command sequence. An Erase Setup

command (20H) is first written to the Command User Interface, followed by the Erase Confirm command (D0H). These commands require both appropriate sequencing and an address within the block to be erased to FFH. Block preconditioning, erase and verify are all handled internally by the Write State Machine, invisible to the system. After the two-command erase sequence is written to it, the 28F008SA automatically outputs Status Register data when read (see Figure 8; Block Erase Flowchart). The CPU can detect the completion of the erase event by analyzing the output of the RY/BY# pin, or the WSM Status bit of the Status Register.

When erase is completed, the Erase Status bit should be checked. If erase error is detected, the Status Register should be cleared. The Command User Interface remains in Read Status Register mode until further commands are issued to it.

This two-step sequence of set-up followed by execution ensures that memory contents are not accidentally erased. Also, reliable block erasure can only occur when $V_{pp} = V_{pph}$. In the absence of this high voltage, memory contents are protected against erasure. If block erase is attempted while $V_{pp} = V_{ppl}$, the V_{pp} Status bit will be set to "1". Erase attempts while $V_{ppl} < V_{pp} < V_{pph}$ produce spurious results and should not be attempted.

Erase Suspend/Erase Resume Commands

The Erase Suspend command allows block erase interruption in order to read data from another block of memory. Once the erase process starts, writing the Erase Suspend command (B0H) to the Command User Interface requests that the WSM suspend the erase sequence at a predetermined point in the erase algorithm. The 28F008SA continues to output Status Register data when read, after the Erase Suspend command is written to it. Polling the WSM Status and Erase Suspend Status bits will determine when the erase operation has been suspended (both will be set to "1"). RY/BY# will also transition to V_{OH} .

At this point, a Read Array command can be written to the Command User Interface to read data from blocks other than that which is suspended. The only other valid commands at this time are Read Status Register (70H) and Erase Resume (D0H), at which time the WSM will continue with the erase process. The Erase Suspend Status and WSM Status bits of the Status Register will be automatically cleared and RY/BY# will return to V_{OL} . After the Erase Resume command is written to it, the 28F008SA automatically outputs Status Register data when read (see Figure 9; Erase Suspend/Resume Flowchart). V_{pp} must remain at V_{pph} while the 28F008SA is in Erase Suspend.

Byte Write Setup/Write Commands (40H or 10H)

Byte write is executed by a two-command sequence. The Byte Write Setup command (40H or 10H) is written to the Command User Interface, followed by a second write specifying the address and data (latched on the rising edge of WE#) to be written. The WSM then takes over, controlling the byte write and write verify algorithms internally. After the two-command byte write sequence is written to it, the 28F008SA automatically outputs Status Register data when read (see Figure 7; Byte Write Flowchart). The CPU can detect the completion of the byte write event by analyzing the output of the RY/BY# pin, or the WSM Status bit of the Status Register. Only the Read Status Register command is valid while byte write is active.

When byte write is complete, the Byte Write Status bit should be checked. If byte write error is detected, the Status Register should be cleared. The internal WSM verify only detects errors for "1"s that do not successfully write to "0"s. The Command User Interface remains in Read Status Register mode until further commands are issued to it. If byte write is attempted while $V_{PP} = V_{PPL}$, the V_{PP} Status bit will be set to "1". Byte write attempts while $V_{PPL} < V_{PP} < V_{PPH}$ produce spurious results and should not be attempted.

EXTENDED BLOCK ERASE/BYTE WRITE CYCLING

Intel has designed extended cycling capability into its ETOX flash memory technologies. The 28F008SA is designed for 100,000 byte write/block erase cycles on each of the sixteen 64-Kbyte blocks. Low electric fields, advanced oxides and minimal oxide area per cell subjected to the tunneling electric field combine to greatly reduce oxide stress and the probability of failure. A 20-Mbyte solid-state drive using an array of 28F008SAs has a MTBF (Mean Time Between Failure) of 33.3 million hours⁽¹⁾, over 600 times more reliable than equivalent rotating disk technology.

AUTOMATED BYTE WRITE

The 28F008SA integrates the Quick-Pulse programming algorithm of prior Intel Flash devices on-chip, using the Command User Interface, Status Register and Write State Machine (WSM). On-chip integration dramatically simplifies system software and provides processor interface timings to the Command User Interface and Status Register. WSM operation, internal verify and V_{PP} high voltage presence are monitored and reported via the RY/BY# output and appropriate Status Register bits. Figure 7 shows a

system software flowchart for device byte write. The entire sequence is performed with V_{PP} at V_{PPH} . Byte write abort occurs when RP# transitions to V_{IL} or V_{PP} drops to V_{PPL} . Although the WSM is halted, byte data is partially written at the location where byte write was aborted. Block erasure, or a repeat of byte write, is required to initialize this data to a known value.

AUTOMATED BLOCK ERASE

As above, the Quick-Erase algorithm of prior Intel Flash devices is now implemented internally, including all preconditioning of block data. WSM operation, erase success and V_{PP} high voltage presence are monitored and reported through RY/BY# and the Status Register. Additionally, if a command other than Erase Confirm is written to the device following Erase Setup, both the Erase Status and Byte Write Status bits will be set to "1"s. When issuing the Erase Setup and Erase Confirm commands, they should be written to an address within the address range of the block to be erased. Figure 8 shows a system software flowchart for block erase.

Erase typically takes 1.6 seconds per block. The Erase Suspend/Erase Resume command sequence allows suspension of this erase operation to read data from a block other than that in which erase is being performed. A system software flowchart is shown in Figure 9.

The entire sequence is performed with V_{PP} at V_{PPH} . Abort occurs when RP# transitions to V_{IL} or V_{PP} falls to V_{PPL} , while erase is in progress. Block data is partially erased by this operation, and a repeat of erase is required to obtain a fully erased block.

DESIGN CONSIDERATIONS

Three-Line Output Control

The 28F008SA will often be used in large memory arrays. Intel provides three control inputs to accommodate multiple memory connections. Three-line control provides for:

- lowest possible memory power dissipation
- complete assurance that data bus contention will not occur

To efficiently use these control inputs, an address decoder should enable CE#, while OE# should be connected to all memory devices and the system's READ# control line. This assures that only selected memory devices have active outputs while deselected memory devices are in Standby Mode. RP# should be connected to the system Powergood signal to prevent unintended writes during system power transitions. Powergood should also toggle during system reset.

⁽¹⁾Assumptions: 10-Kbyte file written every 10 minutes. (20-Mbyte array)/(10-Kbyte file) = 2,000 file writes before erase required.

(2000 files writes/erase) × (100,000 cycles per 28F008SA block) = 200 million file writes.

(200 × 10⁶ file writes) × (10 min/write) × (1 hr/60 min) = 33.3 × 10⁶ MTBF.

RY/BY# and Byte Write/Block Erase Polling

RY/BY# is a full CMOS output that provides a hardware method of detecting byte write and block erase completion. It transitions low time t_{WHRL} after a write or erase command sequence is written to the

28F008SA, and returns to V_{OH} when the WSM has finished executing the internal algorithm.

RY/BY# can be connected to the interrupt input of the system CPU or controller. It is active at all times, not tristated if the 28F008SA CE# or OE# inputs are brought to V_{IH} . RY/BY# is also V_{OH} when the device is in Erase Suspend or deep powerdown modes.

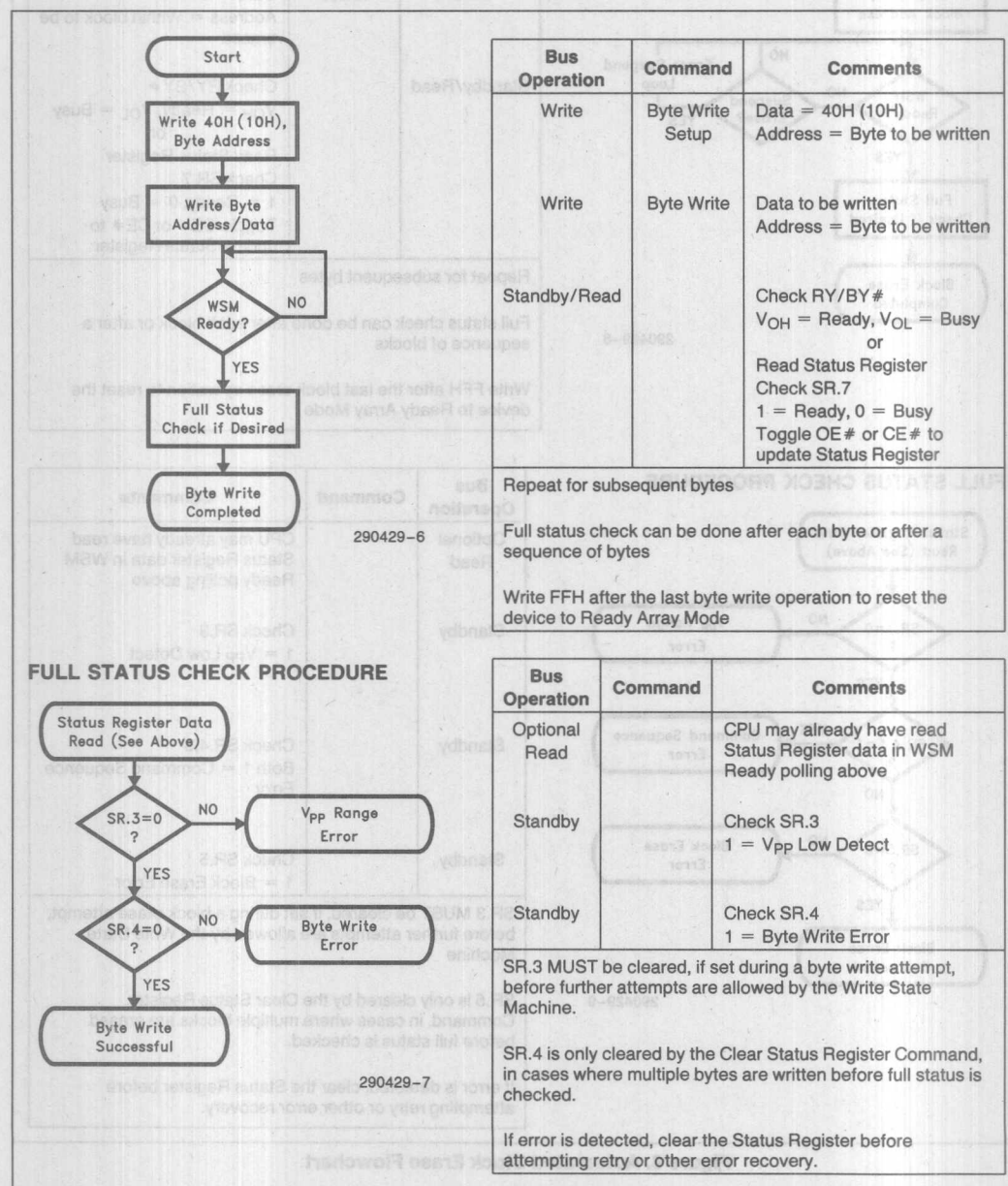
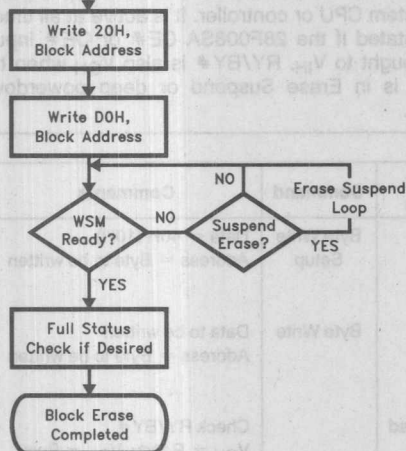


Figure 7. Automated Byte Write Flowchart

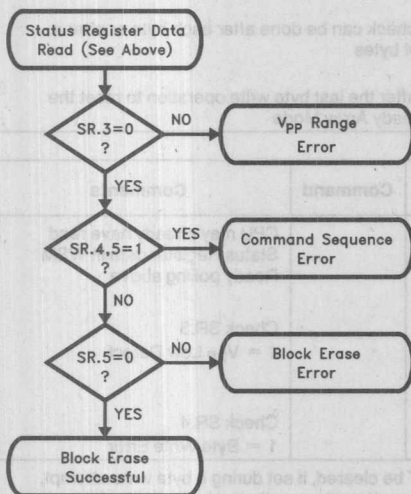
28F008SA



290429-8

Operation	Command	Comments
Write	Erase Setup	Data = 20H Address = Within block to be erased
Write	Erase	Data = D0H Address = Within block to be erased
Standby/Read		Check RY/BY # VOH = Ready, VOL = Busy or Read Status Register Check SR.7 1 = Ready, 0 = Busy Toggle OE# or CE# to update Status Register
Repeat for subsequent bytes		
Full status check can be done after each block or after a sequence of blocks		
Write FFH after the last block erase operation to reset the device to Ready Array Mode		

FULL STATUS CHECK PROCEDURE



290429-9

Bus Operation	Command	Comments
Optional Read		CPU may already have read Status Register data in WSM Ready polling above
Standby		Check SR.3 1 = Vpp Low Detect
Standby		Check SR.4,5 Both 1 = Command Sequence Error
Standby		Check SR.5 1 = Block Erase Error
SR.3 MUST be cleared, if set during a block erase attempt, before further attempts are allowed by the Write State Machine		
SR.5 is only cleared by the Clear Status Register Command, in cases where multiple blocks are erased before full status is checked.		
If error is detected, clear the Status Register before attempting retry or other error recovery.		

Figure 8. Automated Block Erase Flowchart

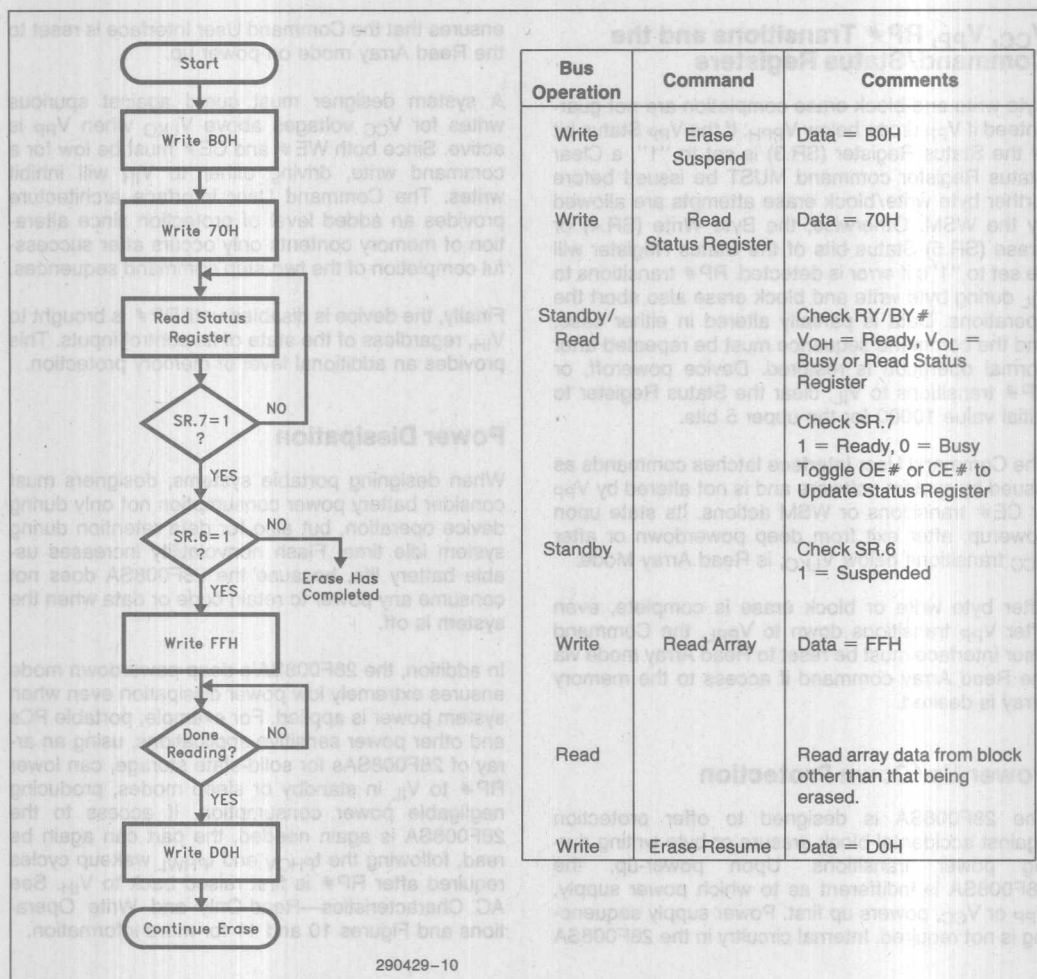


Figure 9. Erase Suspend/Resume Flowchart

Power Supply Decoupling

Flash memory power switching characteristics require careful device decoupling. System designers are interested in 3 supply current issues; standby current levels (I_{SB}), active current levels (I_{CC}) and transient peaks produced by falling and rising edges of CE#. Transient current magnitudes depend on the device outputs' capacitive and inductive loading. Two-line control and proper decoupling capacitor selection will suppress transient voltage peaks. Each device should have a 0.1 μ F ceramic capacitor connected between each VCC and GND, and between its Vpp and GND. These high frequency, low inherent-inductance capacitors should be placed as close as possible to package leads. Additionally, for

every 8 devices, a 4.7 μ F electrolytic capacitor should be placed at the array's power supply connection between VCC and GND. The bulk capacitor will overcome voltage slumps caused by PC board trace inductances.

Vpp Trace on Printed Circuit Boards

Writing flash memories, while they reside in the target system, requires that the printed circuit board designer pay attention to the Vpp power supply trace. The Vpp pin supplies the memory cell current for writing and erasing. Use similar trace widths and layout considerations given to the VCC power bus. Adequate Vpp supply traces and decoupling will decrease Vpp voltage spikes and overshoots.

V_{CC}, V_{PP}, RP # Transitions and the Command/Status Registers

Byte write and block erase completion are not guaranteed if V_{PP} drops below V_{PPH} . If the V_{PP} Status bit of the Status Register (SR.3) is set to "1", a Clear Status Register command MUST be issued before further byte write/block erase attempts are allowed by the WSM. Otherwise, the Byte Write (SR.4) or Erase (SR.5) Status bits of the Status Register will be set to "1"s if error is detected. $RP\#$ transitions to V_{IL} during byte write and block erase also abort the operations. Data is partially altered in either case, and the command sequence must be repeated after normal operation is restored. Device poweroff, or $RP\#$ transitions to V_{IL} , clear the Status Register to initial value 10000 for the upper 5 bits.

The Command User Interface latches commands as issued by system software and is not altered by V_{PP} or $CE\#$ transitions or WSM actions. Its state upon powerup, after exit from deep powerdown or after V_{CC} transitions below V_{IKO} , is Read Array Mode.

After byte write or block erase is complete, even after V_{PP} transitions down to V_{PPL} , the Command User Interface must be reset to Read Array mode via the Read Array command if access to the memory array is desired.

Power Up/Down Protection

The 28F008SA is designed to offer protection against accidental block erasure or byte writing during power transitions. Upon power-up, the 28F008SA is indifferent as to which power supply, V_{PP} or V_{CC} , powers up first. Power supply sequencing is not required. Internal circuitry in the 28F008SA

ensures that the Command User Interface is reset to the Read Array mode on power up.

A system designer must guard against spurious writes for V_{CC} voltages above V_{LKO} when V_{PP} is active. Since both $WE\#$ and $CE\#$ must be low for a command write, driving either to V_{IH} will inhibit writes. The Command User Interface architecture provides an added level of protection since alteration of memory contents only occurs after successful completion of the two-step command sequences.

Finally, the device is disabled until RP# is brought to V_{IH} , regardless of the state of its control inputs. This provides an additional level of memory protection.

Power Dissipation

When designing portable systems, designers must consider battery power consumption not only during device operation, but also for data retention during system idle time. Flash nonvolatility increases usable battery life, because the 28F008SA does not consume any power to retain code or data when the system is off.

In addition, the 28F008SA's deep powerdown mode ensures extremely low power dissipation even when system power is applied. For example, portable PCs and other power sensitive applications, using an array of 28F008SAs for solid-state storage, can lower $RP\#$ to V_{IL} in standby or sleep modes, producing negligible power consumption. If access to the 28F008SA is again needed, the part can again be read, following the t_{PHQV} and t_{PHWL} wakeup cycles required after $RP\#$ is first raised back to V_{IH} . See AC Characteristics—Read-Only and Write Operations and Figures 10 and 11 for more information.

ABSOLUTE MAXIMUM RATINGS*

Operating Temperature	
During Read	0°C to +70°C(1)
During Block Erase/Byte Write	0°C to +70°C
Temperature Under Bias	–10°C to +80°C
Storage Temperature	–65°C to +125°C
Voltage on Any Pin (except V _{CC} and V _{PP}) with Respect to GND	–2.0V to +7.0V(2)
V _{PP} Program Voltage with Respect to GND during Block Erase/Byte Write	–2.0V to +14.0V(2, 3)
V _{CC} Supply Voltage with Respect to GND	–2.0V to +7.0V(2)
Output Short Circuit Current	100 mA(4)

NOTICE: This data sheet contains preliminary information on new products in production. The specifications are subject to change without notice. Verify with your local Intel Sales office that you have the latest data sheet before finalizing a design.

***WARNING:** Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.

NOTES:

- Operating temperature is for commercial product defined by this specification.
- Minimum DC voltage is –0.5V on input/output pins. During transitions, this level may undershoot to –2.0V for periods <20 ns. Maximum DC voltage on input/output pins is V_{CC} + 0.5V which, during transitions, may overshoot to V_{CC} + 2.0V for periods <20 ns.
- Maximum DC voltage on V_{PP} may overshoot to +14.0V for periods <20 ns.
- Output shorted for no more than one second. No more than one output shorted at a time.
- 5% V_{CC} specifications reference the 28F008SA-85 in its High Speed configuration. 10% V_{CC} specifications reference the 28F008SA-85 in its Standard configuration, and the 28F008SA-120.

OPERATING CONDITIONS

Symbol	Parameter	Notes	Min	Max	Unit
T _A	Operating Temperature		0	70	°C
V _{CC}	V _{CC} Supply Voltage (10%)	5	4.50	5.50	V
V _{CC}	V _{CC} Supply Voltage (5%)	5	4.75	5.25	V

DC CHARACTERISTICS

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Condition
I _{LI}	Input Load Current	1			±1.0	μA	V _{CC} = V _{CC} Max V _{IN} = V _{CC} or GND
I _{LO}	Output Leakage Current	1			±10	μA	V _{CC} = V _{CC} Max V _{OUT} = V _{CC} or GND
I _{CCS}	V _{CC} Standby Current	1, 3		1.0	2.0	mA	V _{CC} = V _{CC} Max CE# = RP# = V _{IH}
				30	100	μA	V _{CC} = V _{CC} Max CE# = RP# = V _{CC} ±0.2V
I _{CCD}	V _{CC} Deep PowerDown Current	1		0.20	1.2	μA	RP# = GND ±0.2V I _{OUT} (RY/BY#) = 0 mA
I _{CCR}	V _{CC} Read Current	1		20	35	mA	V _{CC} = V _{CC} Max, CE# = GND f = 8 MHz, I _{OUT} = 0 mA CMOS Inputs
				25	50	mA	V _{CC} = V _{CC} Max, CE# = V _{IL} f = 8 MHz, I _{OUT} = 0 mA TTL Inputs

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Condition
I _{CCW}	V _{CC} Byte Write Current	1		10	30	mA	Byte Write In Progress
I _{CCE}	V _{CC} Block Erase Current	1		10	30	mA	Block Erase In Progress
I _{CCES}	V _{CC} Erase Suspend Current	1, 2		5	10	mA	Block Erase Suspended CE# = V _{IH}
I _{PPS}	V _{PP} Standby Current	1		±1	±15	μA	V _{PP} ≤ V _{CC}
				90	200	μA	V _{PP} > V _{CC}
I _{PPD}	V _{PP} Deep PowerDown Current	1		0.10	5.0	μA	RP# = GND ±0.2V
I _{PPW}	V _{PP} Byte Write Current	1		10	30	mA	V _{PP} = V _{PPH} Byte Write in Progress
I _{PPE}	V _{PP} Block Erase Current	1		10	30	mA	V _{PP} = V _{PPH} Block Erase in Progress
I _{PPES}	V _{PP} Erase Suspend Current	1		90	200	μA	V _{PP} = V _{PPH} Block Erase Suspended
V _{IL}	Input Low Voltage		-0.5		0.8	V	
V _{IH}	Input High Voltage		2.0		V _{CC} + 0.5	V	
V _{OL}	Output Low Voltage	3			0.45	V	V _{CC} = V _{CC} Min I _{OL} = 5.8 mA
V _{OH}	Output High Voltage	3	2.4			V	V _{CC} = V _{CC} Min I _{OH} = -2.5 mA
V _{PPL}	V _{PP} during Normal Operations	4	0.0		6.5	V	
V _{PPH}	V _{PP} during Erase/Write Operations		11.4	12.0	12.6	V	
V _{LKO}	V _{CC} Erase/Write Lock Voltage		2.0			V	

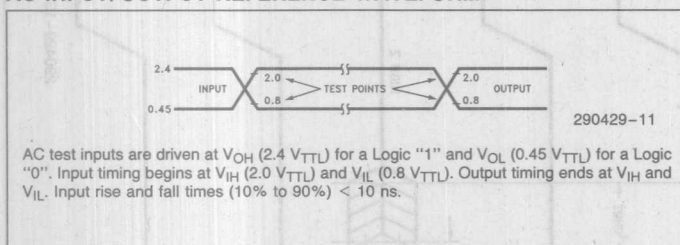
CAPACITANCE(5) T_A = 25°C, f = 1 MHz

Symbol	Parameter	Typ	Max	Unit	Condition
C _{IN}	Input Capacitance	6	8	pF	V _{IN} = 0V
C _{OUT}	Output Capacitance	8	12	pF	V _{OUT} = 0V

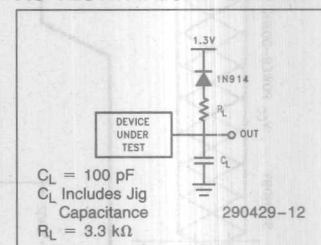
NOTES:

1. All currents are in RMS unless otherwise noted. Typical values at V_{CC} = 5.0V, V_{PP} = 12.0V, T = 25°C. These currents are valid for all product versions (packages and speeds).
2. I_{CCES} is specified with the device deselected. If the 28F008SA is read while in Erase Suspend Mode, current draw is the sum of I_{CCES} and I_{CCR}.
3. Includes RY/BY#.
4. Block Erases/Byte Writes are inhibited when V_{PP} = V_{PPL} and not guaranteed in the range between V_{PPH} and V_{PPL}.
5. Sampled, not 100% tested.

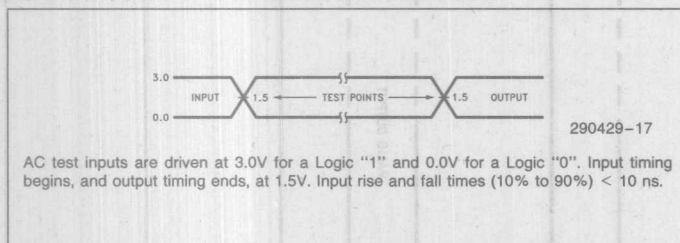
AC INPUT/OUTPUT REFERENCE WAVEFORM(1)



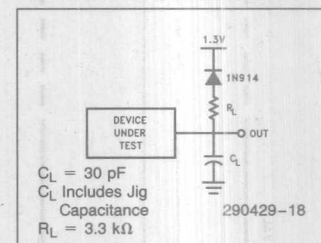
AC TESTING LOAD CIRCUIT(1)



HIGH SPEED AC INPUT/OUTPUT REFERENCE WAVEFORM(2)



HIGH SPEED AC TESTING LOAD CIRCUIT(2)



NOTES:

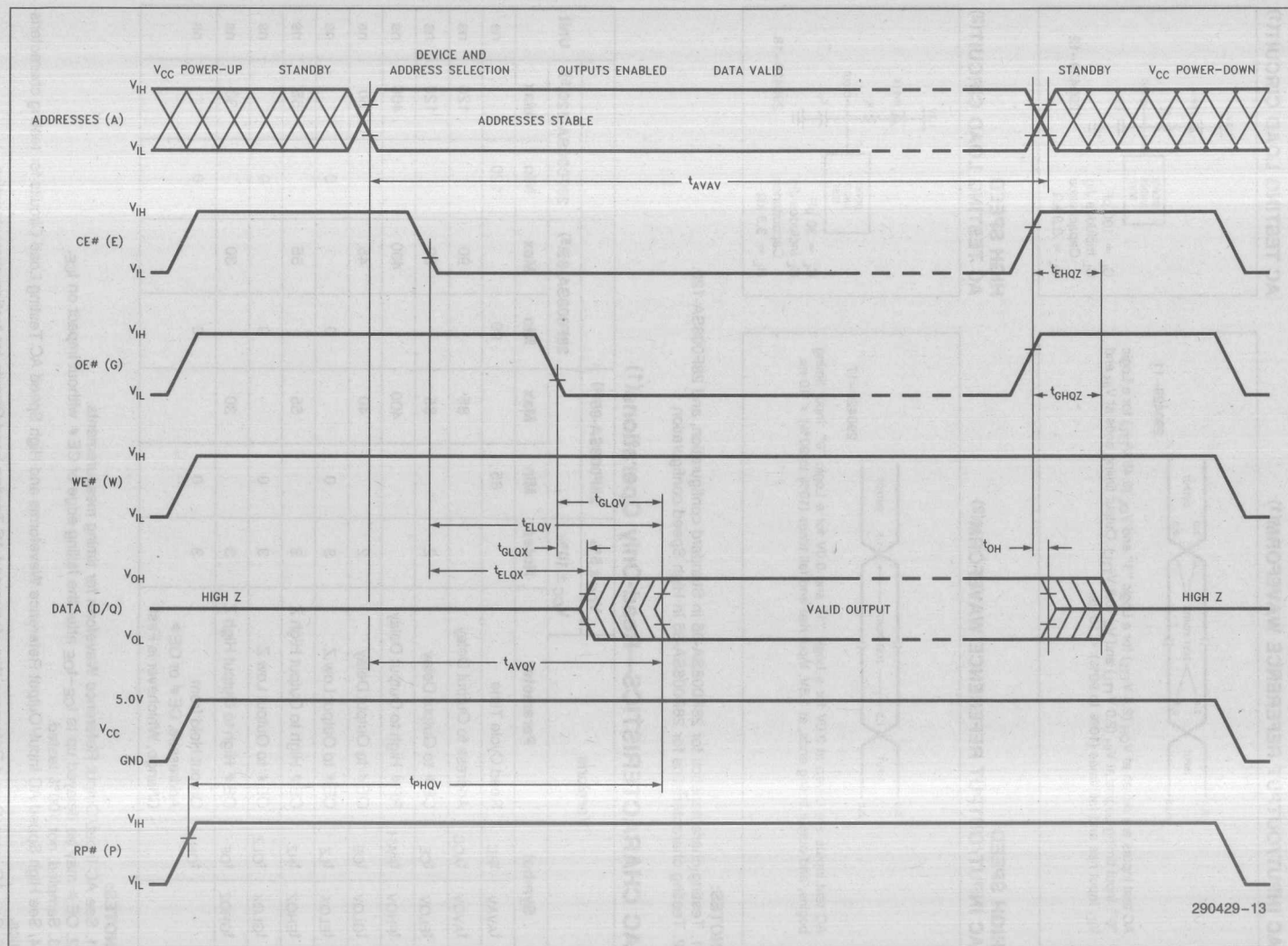
1. Testing characteristics for 28F008SA-85 in Standard configuration, and 28F008SA-120.
2. Testing characteristics for 28F008SA-85 in High Speed configuration.

AC CHARACTERISTICS—Read-Only Operations(1)

Versions		Notes	$V_{CC} \pm 5\%$ 28F008SA-85(4)		$V_{CC} \pm 10\%$ 28F008SA-85(5)		28F008SA-120(5)		Unit
			Min	Max	Min	Max	Min	Max	
t_{AVAV}	t_{RC}	Read Cycle Time	85		90		120		ns
t_{AVQV}	t_{ACC}	Address to Output Delay		85		90		120	ns
t_{ELQV}	t_{CE}	CE# to Output Delay	2	85		90		120	ns
t_{PHQV}	t_{PWH}	RP# High to Output Delay		400		400		400	ns
t_{GLQV}	t_{OE}	OE# to Output Delay	2	40		45		50	ns
t_{ELQX}	t_{LZ}	CE# to Output Low Z	3	0	0		0		ns
t_{EHQZ}	t_{HZ}	CE# High to Output High Z	3	55		55		55	ns
t_{GLQX}	t_{OLZ}	OE# to Output Low Z	3	0	0		0		ns
t_{GHQZ}	t_{DF}	OE# High to Output High Z	3	30		30		30	ns
	t_{OH}	Output Hold from Addresses, CE# or OE# Change, Whichever is First	3	0	0		0		ns

NOTES:

1. See AC Input/Output Reference Waveform for timing measurements.
2. OE# may be delayed up to $t_{CE} - t_{OE}$ after the falling edge of CE# without impact on t_{CE} .
3. Sampled, not 100% tested.
4. See High Speed AC Input/Output Reference Waveforms and High Speed AC Testing Load Circuits for testing characteristics.
5. See AC Input/Output Reference Waveforms and AC Testing Load Circuits for testing characteristics.



AC CHARACTERISTICS—Write Operations(1)

Versions			V _{CC} ± 5%	28F008SA-85 ⁽⁷⁾						Unit
			V _{CC} ± 10%			28F008SA-85 ⁽⁸⁾		28F008SA-120 ⁽⁸⁾		
Symbol		Parameter	Notes	Min	Max	Min	Max	Min	Max	
t _{AVAV}	t _{WC}	Write Cycle Time		85		90		120		ns
t _{PHWL}	t _{PS}	RP# High Recovery to WE# Going Low	2	1		1		1		μs
t _{ELWL}	t _{CS}	CE# Setup to WE# Going Low		10		10		10		ns
t _{WLWH}	t _{WP}	WE# Pulse Width		40		40		40		ns
t _{VPWH}	t _{VPS}	V _{PP} Setup to WE# Going High	2	100		100		100		ns
t _{AVWH}	t _{AS}	Address Setup to WE# Going High	3	40		40		40		ns
t _{DVWH}	t _{DS}	Data Setup to WE# Going High	4	40		40		40		ns
t _{WHDX}	t _{DH}	Data Hold from WE# High		5		5		5		ns
t _{WHAX}	t _{AH}	Address Hold from WE# High		5		5		5		ns
t _{WHEH}	t _{CH}	CE# Hold from WE# High		10		10		10		ns
t _{WHWL}	t _{WPH}	WE# Pulse Width High		30		30		30		ns
t _{WHRL}		WE# High to RY/BY# Going Low			100		100		100	ns
t _{WHQV1}		Duration of Byte Write Operation	5, 6	6		6		6		μs
t _{WHQV2}		Duration of Block Erase Operation	5, 6	0.3		0.3		0.3		sec
t _{WHGL}		Write Recovery before Read		0		0		0		μs
t _{QVVL}	t _{VPH}	V _{PP} Hold from Valid SRD, RY/BY# High	2, 6	0		0		0		ns

NOTES:

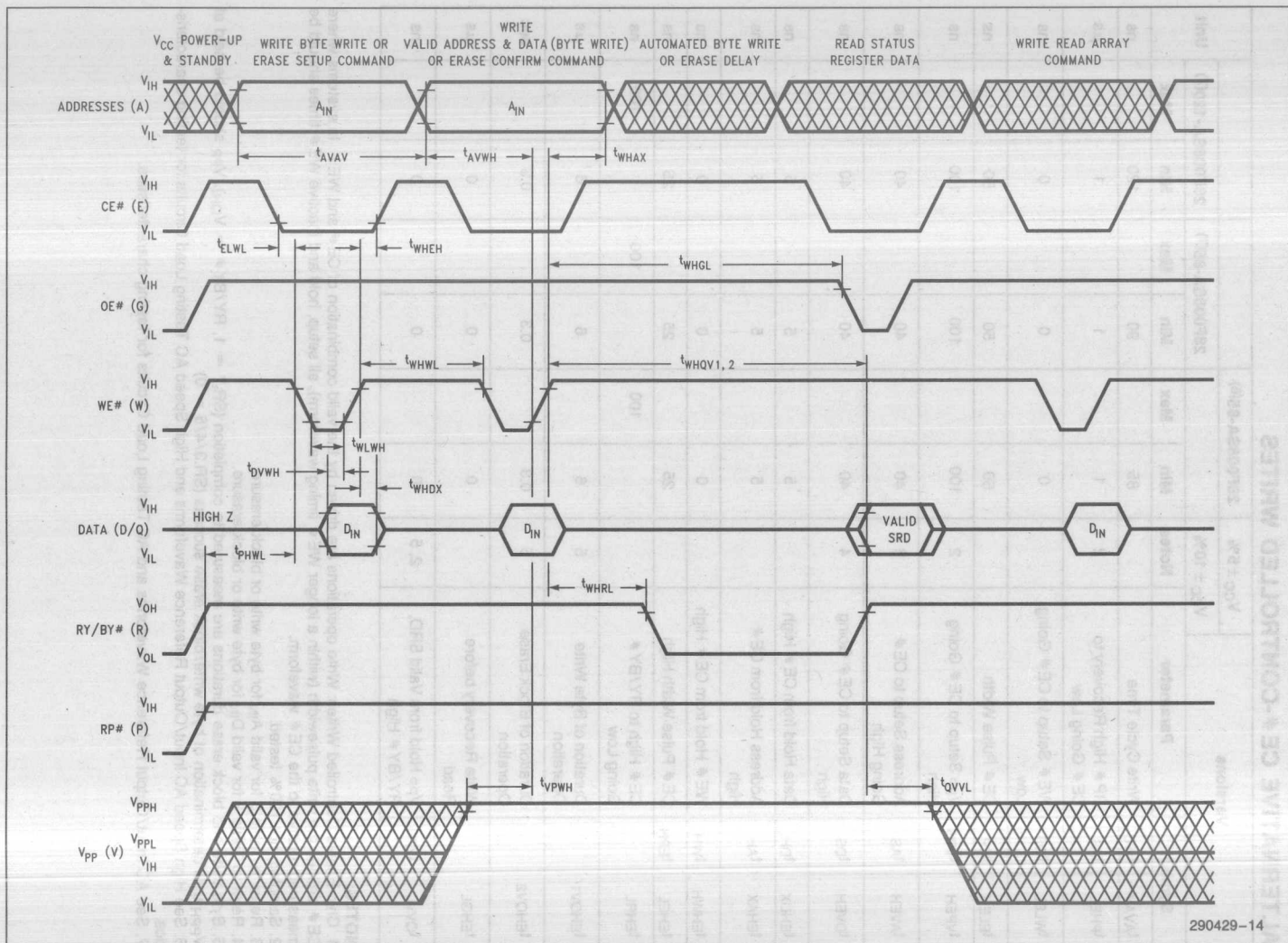
- Read timing characteristics during erase and byte write operations are the same as during read-only operations. Refer to AC Characteristics for Read-Only Operations.
- Sampled, not 100% tested.
- Refer to Table 3 for valid A_{IN} for byte write or block erasure.
- Refer to Table 3 for valid D_{IN} for byte write or block erasure.
- The on-chip Write State Machine incorporates all byte write and block erase system functions and overhead of standard Intel flash memory, including byte program and verify (byte write) and block precondition, precondition verify, erase and erase verify (block erase).
- Byte write and block erase durations are measured to completion (SR.7 = 1, RY/BY# = V_{OH}). V_{PP} should be held at V_{PPH} until determination of byte write/block erase success (SR.3/4/5 = 0)
- See High Speed AC Input/Output Reference Waveforms and High Speed AC Testing Load Circuits for testing characteristics.
- See AC Input/Output Reference Waveforms and AC Testing Load Circuits for testing characteristics.

Parameter	Notes	28F008SA-85			28F008SA-120			Unit
		Min	Typ(1)	Max	Min	Typ(1)	Max	
Block Erase Time	2		1.6	10		1.6	10	sec
Block Write Time	2		0.6	2.1		0.6	2.1	sec

NOTES:

1. 25°C, 12.0 V_{pp}.
2. Excludes System-Level Overhead.

8. See AC Input/Output Reference Waveforms and AC Testing Load Circuits for testing characteristics.
7. See High-Speed AC Input/Output Reference Waveforms and High-Speed AC Testing Load Circuits for testing characteristics.
V_{pp} with determination of data retention/erase success (SR 3VALS ~ 0)
6. Byte write and block erase duration are measured to completion (SR 7 ~ 1, RYBY% ~ V_{pp}). V_{pp} should be held at mass voltage (block erase).
5. Intel flash memory, including data program and verify (byte write) and block precondition, precondition verify, erase and
4. The on-chip Write State Machine incorporates all byte write and block erase system functions and overrides all standard
4. Refer to Table 3 for valid D₀ for byte write or block erase.
3. Refer to Table 3 for valid A₀ for byte write or block erase.
2. Sampled not 100% tested.
AC Characteristics for Read-Only Operations.
1. Read timing characteristics during erase and byte write operations are the same as during read-only operations. Refer to



ALTERNATIVE CE #-CONTROLLED WRITES

Versions		Parameter	Notes	V _{CC} ± 5%		28F008SA-85 ⁽⁶⁾				Unit
				V _{CC} ± 10%				28F008SA-85 ⁽⁷⁾	28F008SA-120 ⁽⁷⁾	
Symbol				Min	Max	Min	Max	Min	Max	
t _{AVAV}	t _{WC}	Write Cycle Time		85		90		120		ns
t _{PHL}	t _{PS}	RP # High Recovery to CE # Going Low	2	1		1		1		μs
t _{WLEL}	t _{WS}	WE # Setup to CE # Going Low		0		0		0		ns
t _{ELEH}	t _{CP}	CE # Pulse Width		50		50		50		ns
t _{VPEH}	t _{VPS}	V _{PP} Setup to CE # Going High	2	100		100		100		ns
t _{AVEH}	t _{AS}	Address Setup to CE # Going High	3	40		40		40		ns
t _{DVEH}	t _{DS}	Data Setup to CE # Going High	4	40		40		40		ns
t _{EHDH}	t _{DH}	Data Hold from CE # High		5		5		5		ns
t _{EHAX}	t _{AH}	Address Hold from CE # High		5		5		5		ns
t _{EHWH}	t _{WH}	WE # Hold from CE # High		0		0		0		ns
t _{EHEL}	t _{EPH}	CE # Pulse Width High		25		25		25		ns
t _{EHRL}		CE # High to RY/BY # Going Low			100		100		100	ns
t _{EHQV1}		Duration of Byte Write Operation	5	6		6		6		μs
t _{EHQV2}		Duration of Block Erase Operation	5	0.3		0.3		0.3		sec
t _{EHGL}		Write Recovery before Read		0		0		0		μs
t _{QVVL}	t _{VPH}	V _{PP} Hold from Valid SRD, RY/BY # High	2, 5	0		0		0		ns

NOTES:

1. Chip-Enable Controlled Writes: Write operations are driven by the valid combination of CE # and WE #. In systems where CE # defines the write pulsewidth (within a longer WE # timing waveform), all setup, hold and inactive WE # times should be measured relative to the CE # waveform.
2. Sampled, not 100% tested.
3. Refer to Table 3 for valid A_{IN} for byte write or block erasure.
4. Refer to Table 3 for valid D_{IN} for byte write or block erasure.
5. Byte write and block erase durations are measured to completion (SR.7 = 1, RY/BY # = V_{OH}). V_{PP} should be held at V_{PPH} until determination of byte write/block erase success (SR.3/4/5 = 0).
6. See High Speed AC Input/Output Reference Waveforms and High Speed AC Testing Load Circuits for testing characteristics.
7. See AC Input/Output Reference Waveforms and AC Testing Load Circuits for testing characteristics.

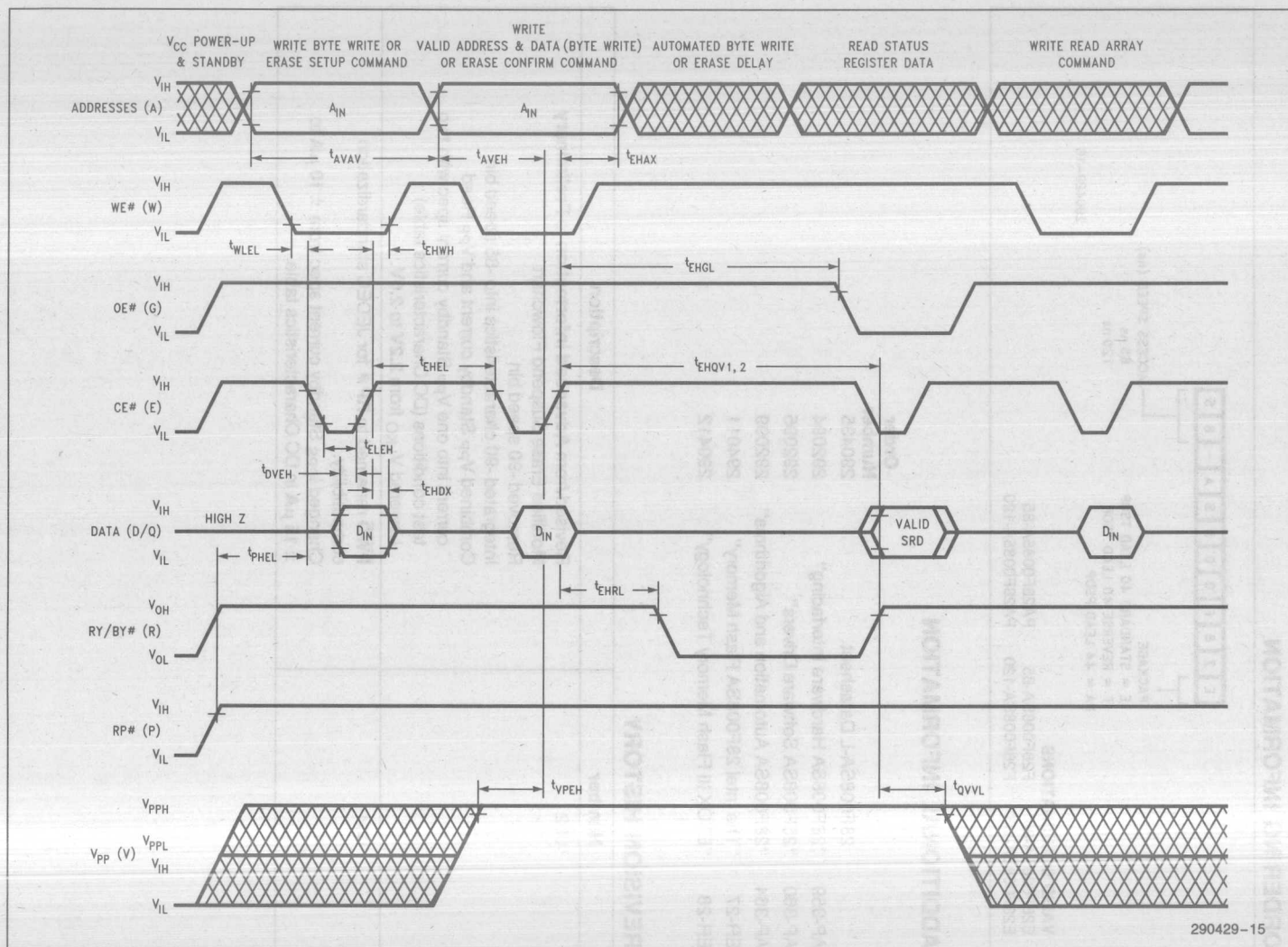


Figure 12. Alternate AC Waveform for Write Operations

PRELIMINARY

E 2 8 F 0 0 8 S A - 8 5

PACKAGE

E = STANDARD 40 LEAD TSOP
F = REVERSE 40 LEAD TSOP
PA = 44 LEAD PSOP

ACCESS SPEED (ns)

85 ns
120 ns

290429-16

VALID COMBINATIONS

E28F008SA-85 F28F008SA-85 PA28F0085A-85
E28F008SA-120 F28F008SA-120 PA28F0085A-120

ADDITIONAL INFORMATION

Order
Number

28F008SA-L Datasheet

290435

AP-359 "28F008SA Hardware Interfacing"

292094

AP-360 "25F008SA Software Drivers"

292095

AP-364 "28F008SA Automation and Algorithms"

292099

ER-27 "The Intel 28F008SA Flash Memory"

294011

ER-28 "ETOX III Flash Memory Technology"

290412

REVISION HISTORY

Number	Description
002	Revised from Advanced Information to Preliminary Modified Erase Suspend Flowchart Removed -90 speed bin Integrated -90 characteristics into -85 speed bin Combined V_{PP} Standby current and V_{PP} Read current into one V_{PP} Standby current spec with two test conditions (DC Characteristics table) Lowered V_{LKO} from 2.2V to 2.0V .
004	PWD renamed to RP # for JEDEC standardization compatibility. Changed I_{PPS} Standby current spec from $\pm 10 \mu A$ to $\pm 15 \mu A$ in DC Characteristics table.

28F008SA-L

8-MBIT (1 MBIT x 8) FLASHFILE™ MEMORY

- **High-Density Symmetrically Blocked Architecture**
 - Sixteen 64-Kbyte Blocks
- **Low-Voltage Operation**
 - $-3.3V \pm 0.3V$ or $5.0V \pm 10\% V_{CC}$
- **Extended Cycling Capability**
 - 10,000 Block Erase Cycles
 - 160,000 Block Erase Cycles per Chip
- **Automated Byte Write and Block Erase**
 - Command User Interface
 - Status Register
- **System Performance Enhancements**
 - RY/BY# Status Output
 - Erase Suspend Capability
- **High-Performance Read**
 - 200 ns Maximum Access Time
- **Deep-Powerdown Mode**
 - $0.20 \mu A I_{CC}$ Typical
- **SRAM-Compatible Write Interface**
- **Hardware Data Protection Feature**
 - Erase/Write Lockout during Power Transitions
- **Industry Standard Packaging**
 - 40-Lead TSOP, 44-Lead PSOP
- **ETOX III Nonvolatile Flash Technology**
 - 12V Byte Write/Block Erase

Intel's 28F008SA-L 8 Mbit FlashFile™ Memory is the highest density nonvolatile read/write solution for solid state storage. The 28F008SA-L's extended cycling, symmetrically blocked architecture, fast access time, write automation and very low power consumption provide a more reliable, lower power, lighter weight and higher performance alternative to traditional rotating disk technology. The 28F008SA-L brings new capabilities to portable computing. Application and operating system software stored in resident flash memory arrays provide instant-on, rapid execute-in-place and protection from obsolescence through in-system software updates. Resident software also extends system battery life and increases reliability by reducing disk drive accesses.

For high density data acquisition applications, the 28F008SA-L offers a more cost-effective and reliable alternative to SRAM and battery. Traditional high density embedded applications, such as telecommunications, can take advantage of the 28F008SA-L's nonvolatility, blocking and minimal system code requirements for flexible firmware and modular software designs.

The 28F008SA-L is offered in 40-lead TSOP (standard and reverse) and 44-lead PSOP packages. Pin assignments simplify board layout when integrating multiple devices in a flash memory array or subsystem. This device uses an integrated Command User Interface and state machine for simplified block erasure and byte write. The 28F008SA-L memory map consists of 16 separately erasable 64-Kbyte blocks.

Intel's 28F008SA-L employs advanced CMOS circuitry for systems requiring low power consumption and noise immunity. Its 200 ns access time provides superior performance when compared with magnetic storage media. A deep powerdown mode lowers power consumption to $0.66 \mu W$ typical thru V_{CC} , crucial in portable computing, handheld instrumentation and other low-power applications. The RP# power control input also provides absolute data protection during system powerup/down.

Manufactured on Intel's 0.8 micron ETOX process, the 28F008SA-L provides the highest levels of quality, reliability and cost-effectiveness.

*Microsoft is a trademark of Microsoft Corporation.

PRODUCT OVERVIEW

The 28F008SA-L is a high-performance **8-Mbit** (8,388,608-bit) memory organized as **1 Mbyte** (1,048,576 bytes) of 8 bits each. **Sixteen 64-Kbyte** (65,536-byte) **blocks** are included on the 28F008SA-L. A memory map is shown in Figure 6 of this specification. A block erase operation erases one of the sixteen blocks of memory in typically **2.0 seconds**, independent of the remaining blocks. Each block can be independently erased and written **10,000 cycles**. **Erase Suspend** mode allows system software to suspend block erase to read data or execute code from any other block of the 28F008SA-L.

The 28F008SA-L is available in the **40-lead TSOP** (Thin Small Outline Package, 1.2 mm thick) and **44-lead PSOP** (Plastic Small Outline) packages. Pin-outs are shown in Figures 2 and 4 of this specification.

The **Command User Interface** serves as the interface between the microprocessor or microcontroller and the internal operation of the 28F008SA-L.

Byte Write and Block Erase Automation allow byte write and block erase operations to be executed using a two-write command sequence to the Command User Interface. The internal **Write State Machine** (WSM) automatically executes the algorithms and timings necessary for byte write and block erase operations, including verifications, thereby unburdening the microprocessor or microcontroller. Writing of memory data is performed in byte increments typically within 11 μ s, **Ipp byte write and block erase currents are 10 mA typical, 30 mA maximum. Vpp byte write and block erase voltage is 11.4V to 12.6V.**

The **Status Register** indicates the status of the WSM and when the WSM successfully completes the desired byte write or block erase operation.

The **RY/BY#** output gives an additional indicator of WSM activity, providing capability for both hardware signal of status (versus software polling) and status masking (interrupt masking for background erase, for example). Status polling using RY/BY# minimizes both CPU overhead and system power consumption. When low, RY/BY# indicates that the WSM is performing a block erase or byte write operation. RY/BY# high indicates that the WSM is ready for new commands, block erase is suspended or the device is in deep powerdown mode.

Maximum access time is **200 ns (t_{ACC})** over the commercial temperature range (0°C to +70°C) and over V_{CC} supply voltage range (3.0V to 3.6V and 4.5V to 5.5V). **Icc active current** (CMOS Read) is **5 mA typical, 12 mA maximum at 5 MHz, 3.3V \pm 0.3V.**

When the CE# and RP# pins are at V_{CC} , the **Icc CMOS Standby** mode is enabled.

A **Deep Powerdown** mode is enabled when the RP# pin is at GND, minimizing power consumption and providing write protection. **Icc current** in deep powerdown is **0.20 μ A typical**. Reset time of 500 ns is required from RP# switching high until outputs are valid to read attempts. Equivalently, the device has a wake time of 1 μ s from RP# high until writes to the Command User Interface are recognized by the 28F008SA-L. With RP# at GND, the WSM is reset and the Status Register is cleared.

290435-1

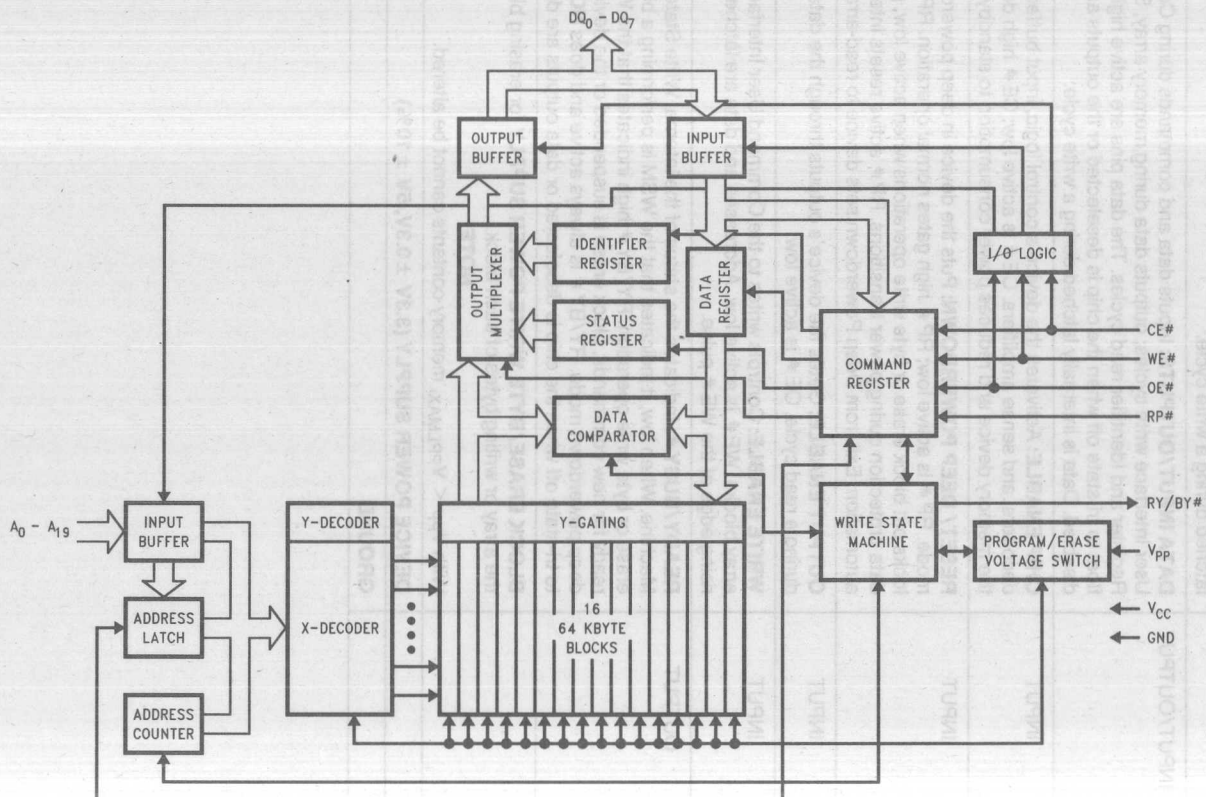


Figure 1. Block Diagram

Symbol	Type	Name and Function
A ₀ –A ₁₉	INPUT	ADDRESS INPUTS for memory addresses. Addresses are internally latched during a write cycle.
DQ ₀ –DQ ₇	INPUT/OUTPUT	DATA INPUT/OUTPUTS: Inputs data and commands during Command User Interface write cycles; outputs data during memory array, Status Register and Identifier read cycles. The data pins are active high and float to tri-state off when the chip is deselected or the outputs are disabled. Data is internally latched during a write cycle.
CE #	INPUT	CHIP ENABLE: Activates the device's control logic, input buffers, decoders, and sense amplifiers. CE # is active low; CE # high deselects the memory device and reduces power consumption to standby levels.
RP #	INPUT	RESET/ DEEP POWERDOWN: Puts the device in deep powerdown mode. RP # is active low; RP # high gates normal operation. RP # also locks out block erase or byte write operations when active low, providing data protection during power transitions. RP # active resets internal automation. Exit from Deep Powerdown sets device to read-array mode.
OE #	INPUT	OUTPUT ENABLE: Gates the device's outputs through the data buffers during a read cycle. OE # is active low.
WE #	INPUT	WRITE ENABLE: Controls writes to the Command User Interface and array blocks. WE # is active low. Addresses and data are latched on the rising edge of the WE # pulse.
RY/BY #	OUTPUT	READY/BUSY #: Indicates the status of the internal Write State Machine. When low, it indicates that the WSM is performing a block erase or byte write operation. RY/BY # high indicates that the WSM is ready for new commands, block erase is suspended or the device is in deep powerdown mode. RY/BY # is always active and does NOT float to tri-state off when the chip is deselected or data outputs are disabled.
V _{PP}		BLOCK ERASE/BYTE WRITE POWER SUPPLY for erasing blocks of the array or writing bytes of each block. NOTE: With V _{PP} < V _{PPLMAX} , memory contents cannot be altered.
V _{CC}		DEVICE POWER SUPPLY (3.3V ± 0.3V, 5V ± 10%)
GND		GROUND

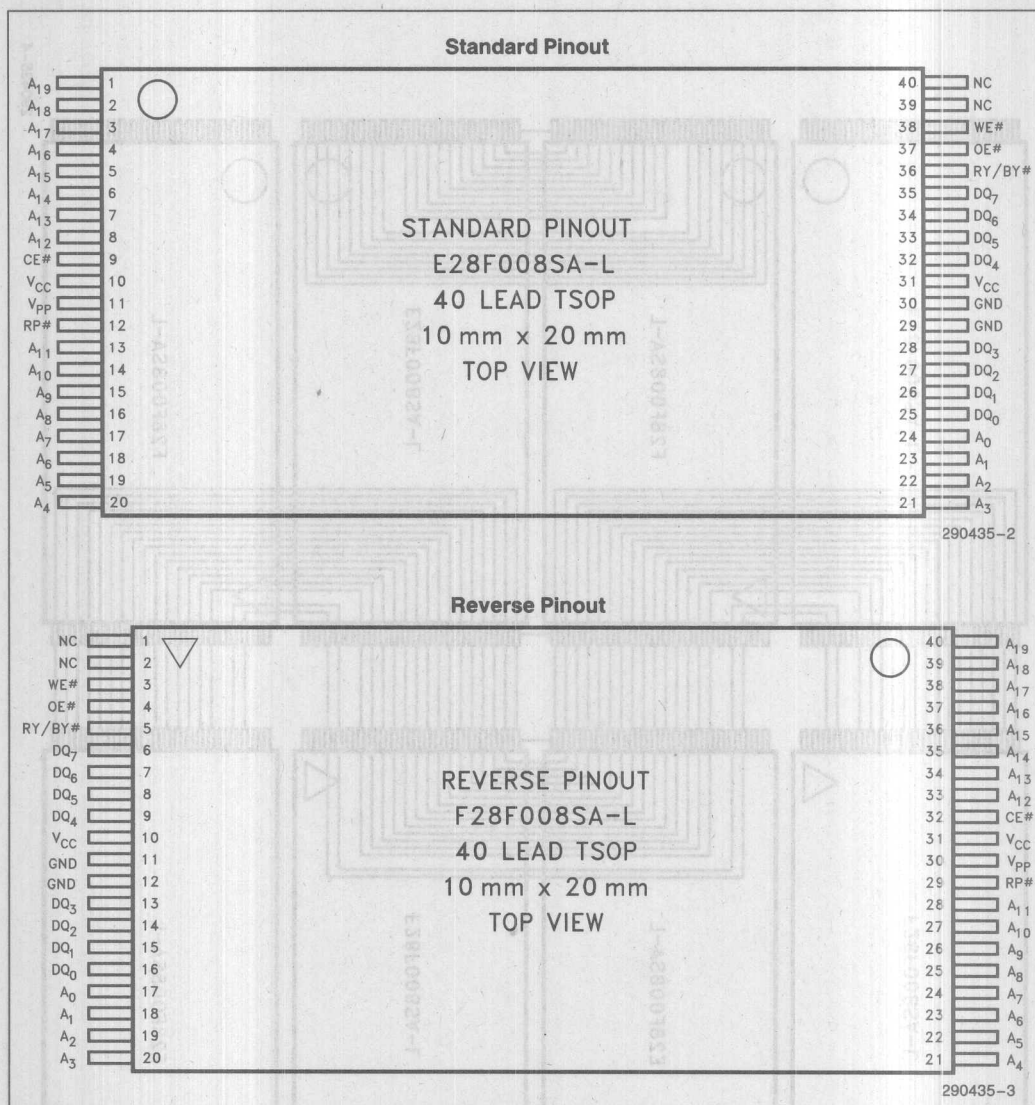


Figure 2. TSOP Lead Configurations

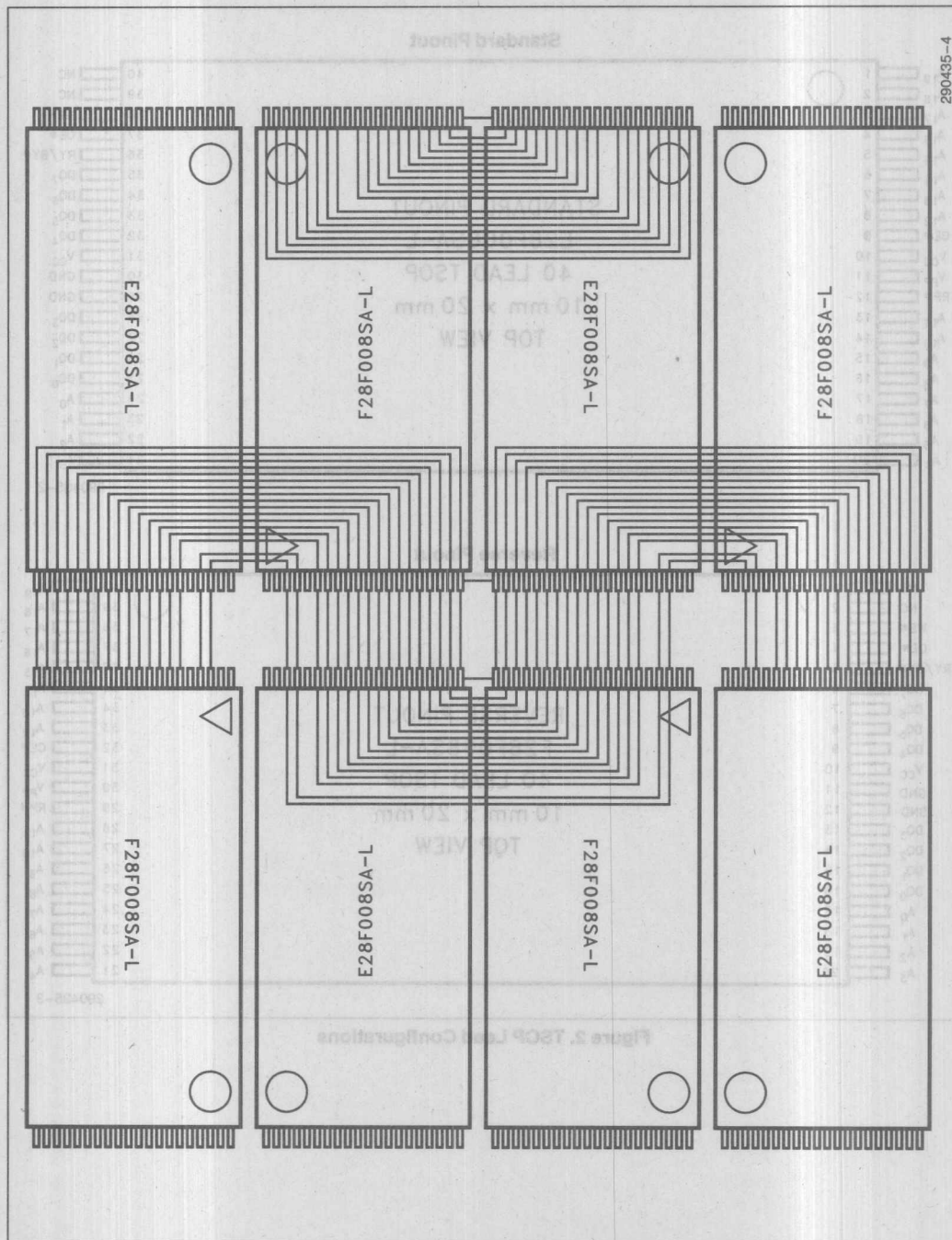


Figure 3. TSOP Serpentine Layout

NOTE:

1. Connect all V_{CC} and GND pins of each device to common power supply outputs: DO NOT leave V_{CC} or GND inputs disconnected.

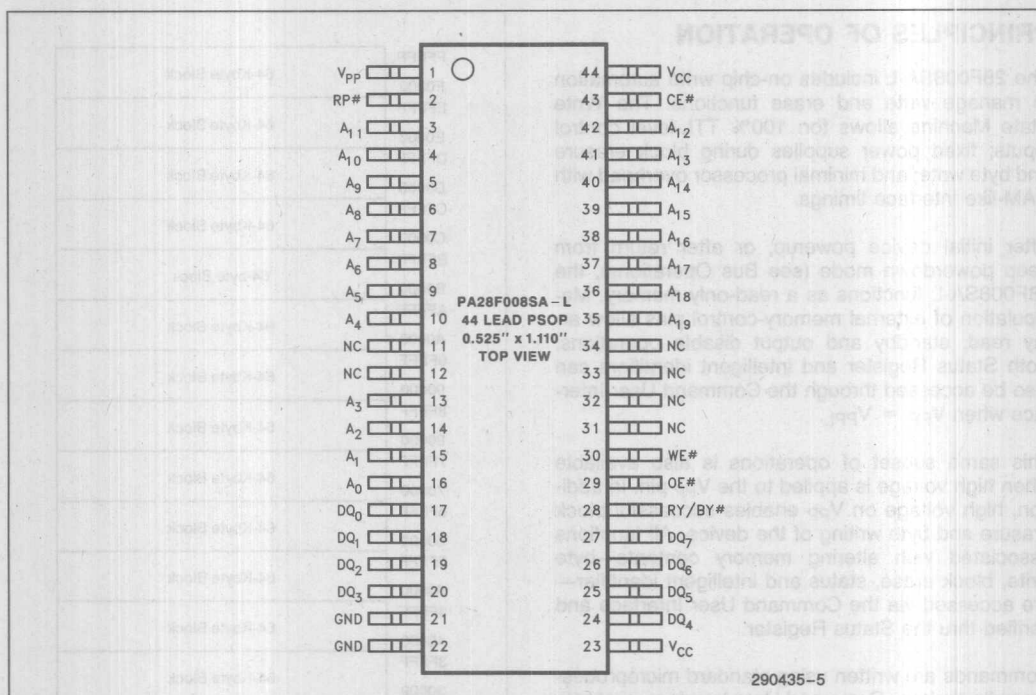


Figure 4. PSOP Lead Configuration

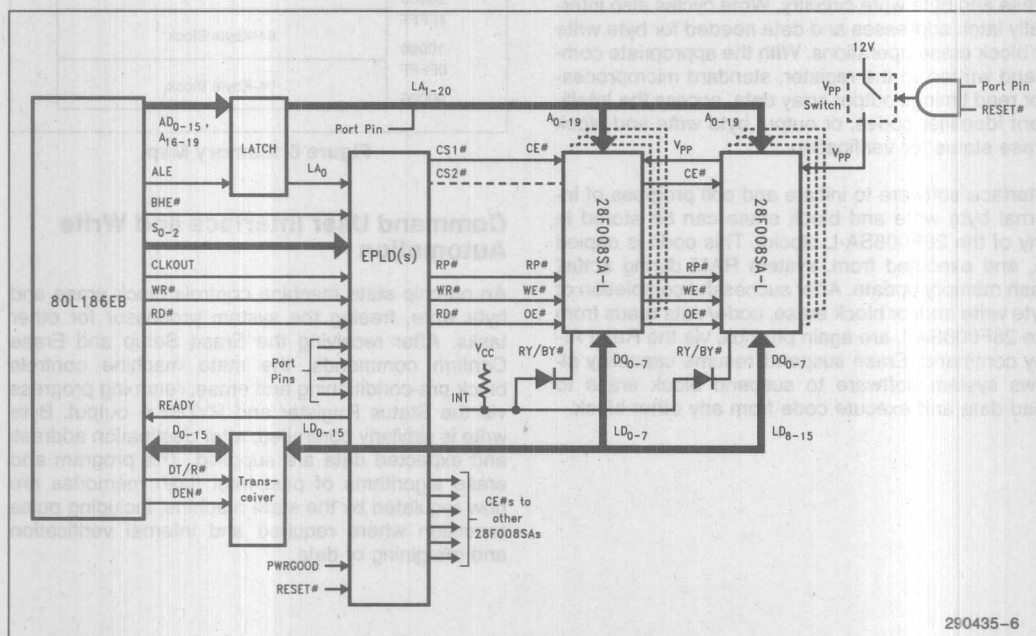


Figure 5. 28F008SA-L Array Interface to Intel 3.3V 80L186EB Embedded Microprocessor

The 28F008SA-L includes on-chip write automation to manage write and erase functions. The Write State Machine allows for: 100% TTL-level control inputs; fixed power supplies during block erasure and byte write; and minimal processor overhead with RAM-like interface timings.

After initial device powerup, or after return from deep powerdown mode (see Bus Operations), the 28F008SA-L functions as a read-only memory. Manipulation of external memory-control pins allow array read, standby and output disable operations. Both Status Register and intelligent identifiers can also be accessed through the Command User Interface when $V_{PP} = V_{PPL}$.

This same subset of operations is also available when high voltage is applied to the V_{PP} pin. In addition, high voltage on V_{PP} enables successful block erasure and byte writing of the device. All functions associated with altering memory contents—byte write, block erase, status and intelligent identifier—are accessed via the Command User Interface and verified thru the Status Register.

Commands are written using standard microprocessor write timings. Command User Interface contents serve as input to the WSM, which controls the block erase and byte write circuitry. Write cycles also internally latch addresses and data needed for byte write or block erase operations. With the appropriate command written to the register, standard microprocessor read timings output array data, access the Intelligent Identifier codes, or output byte write and block erase status for verification.

Interface software to initiate and poll progress of internal byte write and block erase can be stored in any of the 28F008SA-L blocks. This code is copied to, and executed from, system RAM during actual flash memory update. After successful completion of byte write and/or block erase, code/data reads from the 28F008SA-L are again possible via the Read Array command. Erase suspend/resume capability allows system software to suspend block erase to read data and execute code from any other block.

FFFFF	64-Kbyte Block
F0000	
EFFFF	64-Kbyte Block
E0000	
DFFFF	64-Kbyte Block
D0000	
CFFFF	64-Kbyte Block
C0000	
BFFFF	64-Kbyte Block
B0000	
AFFFF	64-Kbyte Block
A0000	
9FFFF	64-Kbyte Block
90000	
8FFFF	64-Kbyte Block
80000	
7FFFF	64-Kbyte Block
70000	
6FFFF	64-Kbyte Block
60000	
5FFFF	64-Kbyte Block
50000	
4FFFF	64-Kbyte Block
40000	
3FFFF	64-Kbyte Block
30000	
2FFFF	64-Kbyte Block
20000	
1FFFF	64-Kbyte Block
10000	
0FFFF	64-Kbyte Block
00000	

Figure 6. Memory Map

Command User Interface and Write Automation

An on-chip state machine controls block erase and byte write, freeing the system processor for other tasks. After receiving the Erase Setup and Erase Confirm commands, the state machine controls block pre-conditioning and erase, returning progress via the Status Register and RY/BY# output. Byte write is similarly controlled, after destination address and expected data are supplied. The program and erase algorithms of past Intel flash memories are now regulated by the state machine, including pulse repetition where required and internal verification and margining of data.

Data Protection

Depending on the application, the system designer may choose to make the V_{pp} power supply switchable (available only when memory byte writes/block erases are required) or hardwired to V_{ppH} . When $V_{pp} = V_{ppL}$, memory contents cannot be altered. The 28F008SA-L Command User Interface architecture provides protection from unwanted byte write or block erase operations even when high voltage is applied to V_{pp} . Additionally, all functions are disabled whenever V_{CC} is below the write lockout voltage V_{LKO} , or when $RP\#$ is at V_{IL} . The 28F008SA-L accommodates either design practice and encourages optimization of the processor-memory interface.

The two-step byte write/block erase Command User Interface write sequence provides additional software write protection.

BUS OPERATION

Flash memory reads, erases and writes in-system via the local CPU. All bus cycles to or from the flash memory conform to standard microprocessor bus cycles.

Read

The 28F008SA-L has three read modes. The memory can be read from any of its blocks, and information can be read from the Intelligent Identifier or Status Register. V_{pp} can be at either V_{ppL} or V_{ppH} .

The first task is to write the appropriate read mode command to the Command User Interface (array, Intelligent Identifier, or Status Register). The 28F008SA-L automatically resets to Read Array mode upon initial device powerup or after exit from deep powerdown. The 28F008SA-L has four control pins, two of which must be logically active to obtain data at the outputs. Chip Enable ($CE\#$) is the device selection control, and when active enables the selected memory device. Output Enable ($OE\#$) is the data input/output (DQ_0 – DQ_7) direction control, and when active drives data from the selected memory onto the I/O bus. $RP\#$ and $WE\#$ must also be at V_{IH} . Figure 10 illustrates read bus cycle waveforms.

Output Disable

With $OE\#$ at a logic-high level (V_{IH}), the device outputs are disabled. Output pins (DQ_0 – DQ_7) are placed in a high-impedance state.

Standby

$CE\#$ at a logic-high level (V_{IH}) places the 28F008SA-L in standby mode. Standby operation disables much of the 28F008SA-L's circuitry and substantially reduces device power consumption. The outputs (DQ_0 – DQ_7) are placed in a high-impedance state independent of the status of $OE\#$. If the 28F008SA-L is deselected during block erase or byte write, the device will continue functioning and consuming normal active power until the operation completes.

Table 2. Bus Operations

Mode	Notes	$RP\#$	$CE\#$	$OE\#$	$WE\#$	A_0	V_{pp}	DQ_0 –7	$RY/BY\#$
Read	1, 2, 3	V_{IH}	V_{IL}	V_{IL}	V_{IH}	X	X	D_{OUT}	X
Output Disable	3	V_{IH}	V_{IL}	V_{IH}	V_{IH}	X	X	High Z	X
Standby	3	V_{IH}	V_{IH}	X	X	X	X	High Z	X
Deep PowerDown		V_{IL}	X	X	X	X	X	High Z	V_{OH}
Intelligent Identifier (Mfr)		V_{IH}	V_{IL}	V_{IL}	V_{IH}	V_{IL}	X	89H	V_{OH}
Intelligent Identifier (Device)		V_{IH}	V_{IL}	V_{IL}	V_{IH}	V_{IH}	X	A1H	V_{OH}
Write	3, 4, 5	V_{IH}	V_{IL}	V_{IH}	V_{IL}	X	X	D_{IN}	X

NOTES:

1. Refer to DC Characteristics. When $V_{pp} = V_{ppL}$, memory contents can be read but not written or erased.
2. X can be V_{IL} or V_{IH} for control pins and addresses, and V_{ppL} or V_{ppH} for V_{pp} . See DC Characteristics for V_{ppL} and V_{ppH} voltages.
3. $RY/BY\#$ is V_{OL} when the Write State Machine is executing internal block erase or byte write algorithms. It is V_{OH} when the WSM is not busy, in Erase Suspend mode or deep powerdown mode.
4. Command writes involving block erase or byte write are only successfully executed when $V_{pp} = V_{ppH}$.
5. Refer to Table 3 for valid D_{IN} during a write operation.

Deep Power-Down

The 28F008SA-L offers a deep power-down feature, entered when RP# is at V_{IL} . Current draw thru V_{CC} is 0.20 μA typical in deep powerdown mode, with current draw through V_{PP} typically 0.1 μA . During read modes, RP#-low deselects the memory, places output drivers in a high-impedance state and turns off all internal circuits. The 28F008SA-L requires time t_{PHQV} (see AC Characteristics-Read-Only Operations) after return from powerdown until initial memory access outputs are valid. After this wakeup interval, normal operation is restored. The Command User Interface is reset to Read Array, and the upper 5 bits of the Status Register are cleared to value 10000, upon return to normal operation.

During block erase or byte write modes, RP# low will abort either operation. Memory contents of the block being altered are no longer valid as the data will be partially written or erased. Time t_{PHWL} after RP# goes to logic-high (V_{IH}) is required before another command can be written.

This use of RP# during system reset is important with automated write/erase devices. When the system come out of reset, it expects to read from the flash memory. Automated flash memories provide

status information when accessed during write/erase modes. If a CPU reset occurs with no flash memory reset, proper CPU initialization would not occur because the flash memory would be providing the status information instead of array data. Intel's Flash Memories allow proper CPU initialization following a system reset through the use of the RP# input. In this application RP# is controlled by the same RESET# signal that resets the system CPU.

Intelligent Identifier Operation

The Intelligent Identifier operation outputs the manufacturer code, 89H; and the device code, A1H for the 28F008SA-L. The system CPU can then automatically match the device with its proper block erase and byte write algorithms.

The manufacturer- and device-codes are read via the Command User Interface. Following a write of 90H to the Command User Interface, a read from address location 00000H outputs the manufacturer code (89H). A read from address 00001H outputs the device code (A1H). It is not necessary to have high voltage applied to V_{PP} to read the Intelligent Identifiers from the Command User Interface.

Table 3. Command Definitions

Command	Bus Cycles Req'd	Notes	First Bus Cycle			Second Bus Cycle		
			Operation	Address	Data	Operation	Address	Data
Read Array/Reset	1	1	Write	X	FFH			
Intelligent Identifier	3	2, 3, 4	Write	X	90H	Read	IA	IID
Read Status Register	2	3	Write	X	70H	Read	X	SRD
Clear Status Register	1		Write	X	50H			
Erase Setup/Erase Confirm	2	2	Write	BA	20H	Write	BA	D0H
Erase Suspend/Erase Resume	2		Write	X	B0H	Write	X	D0H
Byte Write Setup/Write	2	2, 3, 5	Write	WA	40H	Write	WA	WD
Alternate Byte Write Setup/Write	2	2, 3, 5	Write	WA	10H	Write	WA	WD

NOTES:

- Bus operations are defined in Table 2.
- IA = Identifier Address: 00H for manufacturer code, 01H for device code.
BA = Address within the block being erased.
WA = Address of memory location to be written.
- SRD = Data read from Status Register. See Table 4 for a description of the Status Register bits.
WD = Data to be written at location WA. Data is latched on the rising edge of WE#.
IID = Data read from Intelligent Identifiers.
- Following the Intelligent Identifier command, two read operations access manufacture and device codes.
- Either 40H or 10H are recognized by the WSM as the Byte Write Setup command.
- Commands other than those shown above are reserved by Intel for future device implementations and should not be used.

Write

Writes to the Command User Interface enable reading of device data and Intelligent Identifiers. They also control inspection and clearing of the Status Register. Additionally, when $V_{pp} = V_{ppH}$, the Command User Interface controls block erasure and byte write. The contents of the interface register serve as input to the internal state machine.

The Command User Interface itself does not occupy an addressable memory location. The interface register is a latch used to store the command and address and data information needed to execute the command. Erase Setup and Erase Confirm commands require both appropriate command data and an address within the block to be erased. The Byte Write Setup command requires both appropriate command data and the address of the location to be written, while the Byte Write command consists of the data to be written and the address of the location to be written.

The Command User Interface is written by bringing $WE\#$ to a logic-low level (V_{IL}) while $CE\#$ is low. Addresses and data are latched on the rising edge of $WE\#$. Standard microprocessor write timings are used.

Refer to AC Write Characteristics and the AC Waveforms for Write Operations, Figure 11, for specific timing parameters.

COMMAND DEFINITIONS

When V_{pPL} is applied to the V_{pp} pin, read operations from the Status Register, Intelligent Identifiers, or array blocks are enabled. Placing V_{ppH} on V_{pp} enables successful byte write and block erase operations as well.

Device operations are selected by writing specific commands into the Command User Interface. Table 3 defines the 28F008SA-L commands.

Read Array Command

Upon initial device powerup and after exit from deep powerdown mode, the 28F008SA-L defaults to Read Array mode. This operation is also initiated by writing FFH into the Command User Interface. Microprocessor read cycles retrieve array data. The device remains enabled for reads until the Command User Interface contents are altered. Once the internal Write State Machine has started a block erase or byte write operation, the device will not recognize the Read Array command, until the WSM has completed its operation. The Read Array command is functional when $V_{pp} = V_{pPL}$ or V_{ppH} .

Intelligent Identifier Command

The 28F008SA-L contains an Intelligent Identifier operation, initiated by writing 90H into the Command

Table 4. Status Register Definitions

WSMS	ESS	ES	BWS	VPPS	R	R	R
7	6	5	4	3	2	1	0
SR.7 = WRITE STATE MACHINE STATUS							
1 = Ready							
0 = Busy							
SR.6 = ERASE SUSPEND STATUS							
1 = Erase Suspended							
0 = Erase in Progress/Completed							
SR.5 = ERASE STATUS							
1 = Error in Block Erasure							
0 = Successful Block Erase							
SR.4 = BYTE WRITE STATUS							
1 = Error in Byte Write							
0 = Successful Byte Write							
SR.3 = V_{pp} STATUS							
1 = V_{pp} Low Detect; Operation Abort							
0 = V_{pp} OK							
SR.2-SR.0 = RESERVED FOR FUTURE ENHANCEMENTS							
These bits are reserved for future use and should be masked out when polling the Status Register.							

NOTES:

RY/BY# or the Write State Machine Status bit must first be checked to determine byte write or block erase completion, before the Byte Write or Erase Status bit are checked for success.

If the Byte Write AND Erase Status bits are set to "1"s during a block erase attempt, an improper command sequence was entered. Attempt the operation again.

If V_{pp} low status is detected, the Status Register must be cleared before another byte write or block erase operation is attempted.

The V_{pp} Status bit, unlike an A/D converter, does not provide continuous indication of V_{pp} level. The WSM interrogates the V_{pp} level only after the byte write or block erase command sequences have been entered and informs the system if V_{pp} has not been switched on. The V_{pp} Status bit is not guaranteed to report accurate feedback between V_{pPL} and V_{ppH} .

cycle from address 00000H retrieves the manufacturer code of 89H. A read cycle from address 00001H returns the device code of A1H. To terminate the operation, it is necessary to write another valid command into the register. Like the Read Array command, the Intelligent Identifier command is functional when $V_{pp} = V_{pPL}$ or V_{ppH} .

Read Status Register Command

The 28F008SA-L contains a Status Register which may be read to determine when a byte write or block erase operation is complete, and whether that operation completed successfully. The Status Register may be read at any time by writing the Read Status Register command (70H) to the Command User Interface. After writing this command, all subsequent read operations output data from the Status Register, until another valid command is written to the Command User Interface. The contents of the Status Register are latched on the falling edge of OE# or CE#, whichever occurs last in the read cycle. OE# or CE# must be toggled to V_{IH} before further reads to update the Status Register latch. The Read Status Register command functions when $V_{pp} = V_{pPL}$ or V_{ppH} .

Clear Status Register Command

The Erase Status and Byte Write Status bits are set to "1"s by the Write State Machine and can only be reset by the Clear Status Register Command. These bits indicate various failure conditions (see Table 4). By allowing system software to control the resetting of these bits, several operations may be performed (such as cumulatively writing several bytes or erasing multiple blocks in sequence). The Status Register may then be polled to determine if an error occurred during that sequence. This adds flexibility to the way the device may be used.

Additionally, the V_{pp} Status bit (SR.3) MUST be reset by system software before further byte writes or block erases are attempted. To clear the Status Register, the Clear Status Register command (50H) is written to the Command User Interface. The Clear Status Register command is functional when $V_{pp} = V_{pPL}$ or V_{ppH} .

Erase Setup/Erase Confirm Commands

Erase is executed one block at a time, initiated by a two-cycle command sequence. An Erase Setup command (20H) is first written to the Command User

(D0H). These commands require both appropriate sequencing and an address within the block to be erased to FFH. Block preconditioning, erase and verify are all handled internally by the Write State Machine, invisible to the system. After the two-command erase sequence is written to it, the 28F008SA-L automatically outputs Status Register data when read (see Figure 8; Block Erase Flowchart). The CPU can detect the completion of the erase event by analyzing the output of the RY/BY# pin, or the WSM Status bit of the Status Register.

When erase is completed, the Erase Status bit should be checked. If erase error is detected, the Status Register should be cleared. The Command User Interface remains in Read Status Register mode until further commands are issued to it.

This two-step sequence of set-up followed by execution ensures that memory contents are not accidentally erased. Also, reliable block erasure can only occur when $V_{pp} = V_{ppH}$. In the absence of this high voltage, memory contents are protected against erasure. If block erase is attempted while $V_{pp} = V_{pPL}$, the V_{pp} Status bit will be set to "1". Erase attempts while $V_{pPL} < V_{pp} < V_{ppH}$ produce spurious results and should not be attempted.

Erase Suspend/Erase Resume Commands

The Erase Suspend command allows block erase interruption in order to read data from another block of memory. Once the erase process starts, writing the Erase Suspend command (B0H) to the Command User Interface requests that the WSM suspend the erase sequence at a predetermined point in the erase algorithm. The 28F008SA-L continues to output Status Register data when read, after the Erase Suspend command is written to it. Polling the WSM Status and Erase Suspend Status bits will determine when the erase operation has been suspended (both will be set to "1"). RY/BY# will also transition to V_{OH} .

At this point, a Read Array command can be written to the Command User Interface to read data from blocks other than that which is suspended. The only other valid commands at this time are Read Status Register (70H) and Erase Resume (D0H), at which time the WSM will continue with the erase process. The Erase Suspend Status and WSM Status bits of the Status Register will be automatically cleared and RY/BY# will return to V_{OL} . After the Erase Resume command is written to it, the 28F008SA-L automatically outputs Status Register data when read (see Figure 9; Erase Suspend/Resume Flowchart). V_{pp} must remain at V_{ppH} while the 28F008SA-L is in Erase Suspend.

Byte Write Setup/Write Commands (40H or 10H)

Byte write is executed by a two-command sequence. The Byte Write Setup command (40H or 10H) is written to the Command User Interface, followed by a second write specifying the address and data (latched on the rising edge of WE#) to be written. The WSM then takes over, controlling the byte write and write verify algorithms internally. After the two-command byte write sequence is written to it, the 28F008SA-L automatically outputs Status Register data when read (see Figure 7; Byte Write Flowchart). The CPU can detect the completion of the byte write event by analyzing the output of the RY/BY# pin, or the WSM Status bit of the Status Register. Only the Read Status Register command is valid while byte write is active.

When byte write is complete, the Byte Write Status bit should be checked. If byte write error is detected, the Status Register should be cleared. The internal WSM verify only detects errors for "1"s that do not successfully write to "0"s. The Command User Interface remains in Read Status Register mode until further commands are issued to it. If byte write is attempted while $V_{PP} = V_{PPL}$, the V_{PP} Status bit will be set to "1". Byte write attempts while $V_{PPL} < V_{PP} < V_{PPH}$ produce spurious results and should not be attempted.

EXTENDED BLOCK ERASE/BYTE WRITE CYCLING

Intel has designed extended cycling capability into its ETOX flash memory technologies. The 28F008SA-L is designed for 10,000 byte write/block erase cycles on each of the sixteen 64-Kbyte blocks. Low electric fields, advanced oxides and minimal oxide area per cell subjected to the tunneling electric field combine to greatly reduce oxide stress and the probability of failure. A 20-Mbyte solid-state drive using an array of 28F008SA-Ls has a MTBF (Mean Time Between Failure) of 3.33 million hours⁽¹⁾, over 60 times more reliable than equivalent rotating disk technology.

AUTOMATED BYTE WRITE

The 28F008SA-L integrates the Quick-Pulse programming algorithm of prior Intel Flash Memory devices on-chip, using the Command User Interface, Status Register and Write State Machine (WSM). On-chip integration dramatically simplifies system software and provides processor interface timings to the Command User Interface and Status Register. WSM operation, internal verify and V_{PP} high voltage presence are monitored and reported via the RY/BY# output and appropriate Status Register

bits. Figure 7 shows a system software flowchart for device byte write. The entire sequence is performed with V_{PP} at V_{PPH} . Byte write abort occurs when RP# transitions to V_{IL} , or V_{PP} drops to V_{PPL} . Although the WSM is halted, byte data is partially written at the location where byte write was aborted. Block erasure, or a repeat of byte write, is required to initialize this data to a known value.

AUTOMATED BLOCK ERASE

As above, the Quick-Erase algorithm of prior Intel Flash devices is now implemented internally, including all preconditioning of block data. WSM operation, erase success and V_{PP} high voltage presence are monitored and reported through RY/BY# and the Status Register. Additionally, if a command other than Erase Confirm is written to the device following Erase Setup, both the Erase Status and Byte Write Status bits will be set to "1"s. When issuing the Erase Setup and Erase Confirm commands, they should be written to an address within the address range of the block to be erased. Figure 8 shows a system software flowchart for block erase.

Erase typically takes 2.0 seconds per block. The Erase Suspend/Erase Resume command sequence allows suspension of this erase operation to read data from a block other than that in which erase is being performed. A system software flowchart is shown in Figure 9.

The entire sequence is performed with V_{PP} at V_{PPH} . Abort occurs when RP# transitions to V_{IL} or V_{PP} falls to V_{PPL} , while erase is in progress. Block data is partially erased by this operation, and a repeat of erase is required to obtain a fully erased block.

DESIGN CONSIDERATIONS

Three-Line Output Control

The 28F008SA-L will often be used in large memory arrays. Intel provides three control inputs to accommodate multiple memory connections. Three-line control provides for:

- lowest possible memory power dissipation
- complete assurance that data bus contention will not occur

To efficiently use these control inputs, an address decoder should enable CE#, while OE# should be connected to all memory devices and the system's READ# control line. This assures that only selected memory devices have active outputs while deselected memory devices are in Standby Mode. RP# should be connected to the system Powergood signal to prevent unintended writes during system power transitions. Powergood should also toggle during system reset.

⁽¹⁾Assumptions: 10-Kbyte file written every 10 minutes. (20-Mbyte array)/(10-Kbyte file) = 2,000 file writes before erase required.
(2000 files writes/erase) × (10,000 cycles per 28F008SA-L block) = 20 million file writes.
(20 × 10⁶ file writes) × (10 min/write) × (1 hr/60 min) = 3.33 × 10⁶ MTBF.

RY/BY# and Byte Write/Block Erase Polling

RY/BY# is a full CMOS output that provides a hardware method of detecting byte write and block erase completion. It transitions low time t_{WHRL} after a write or erase command sequence is written to the

28F008SA-L, and returns to V_{OH} when the WSM has finished executing the internal algorithm.

RY/BY# can be connected to the interrupt input of the system CPU or controller. It is active at all times, not tristated if the 28F008SA-L CE# or OE# inputs are brought to V_{IH} . RY/BY# is also V_{OH} when the device is in Erase Suspend or deep powerdown modes.

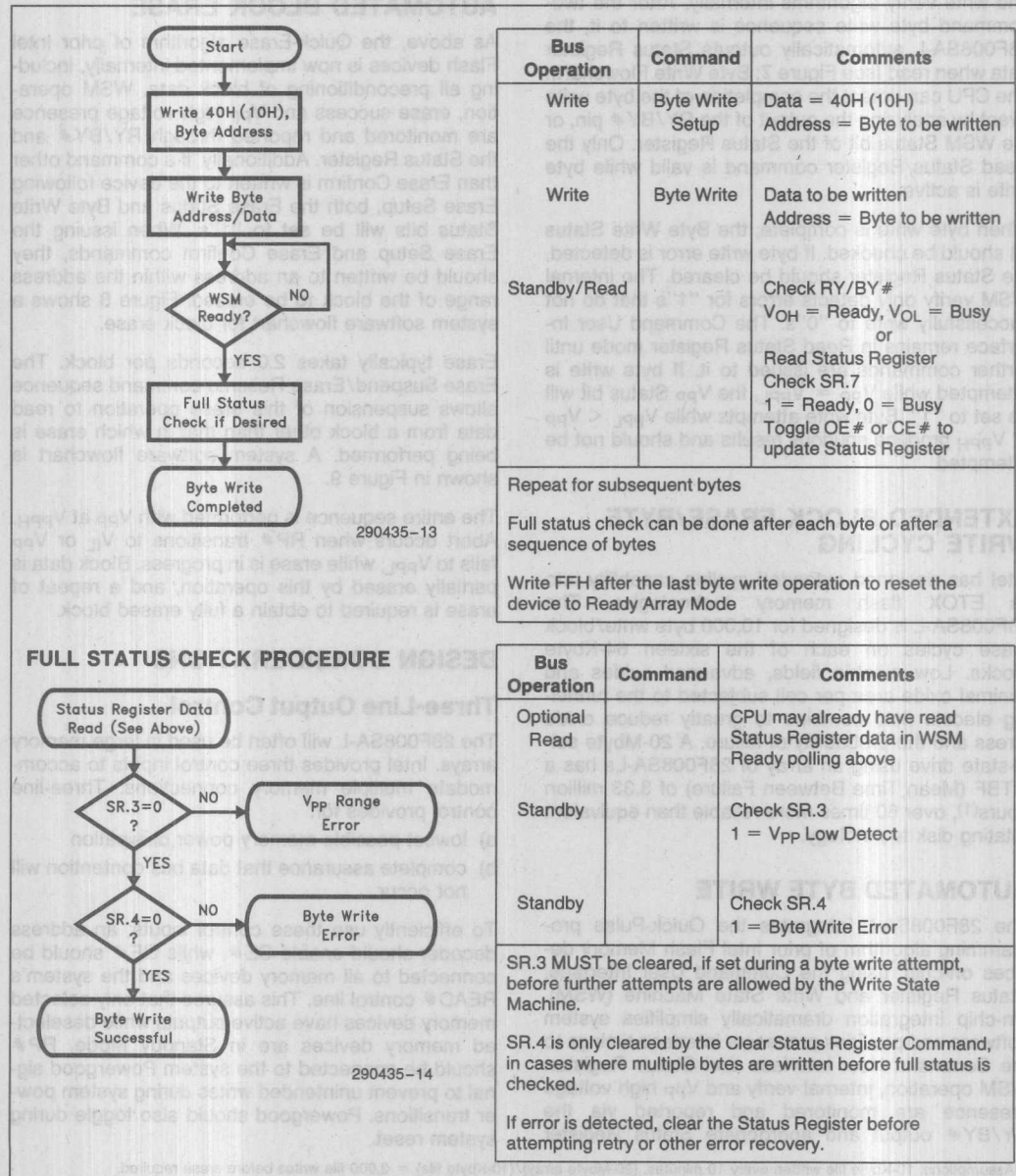


Figure 7. Automated Byte Write Flowchart

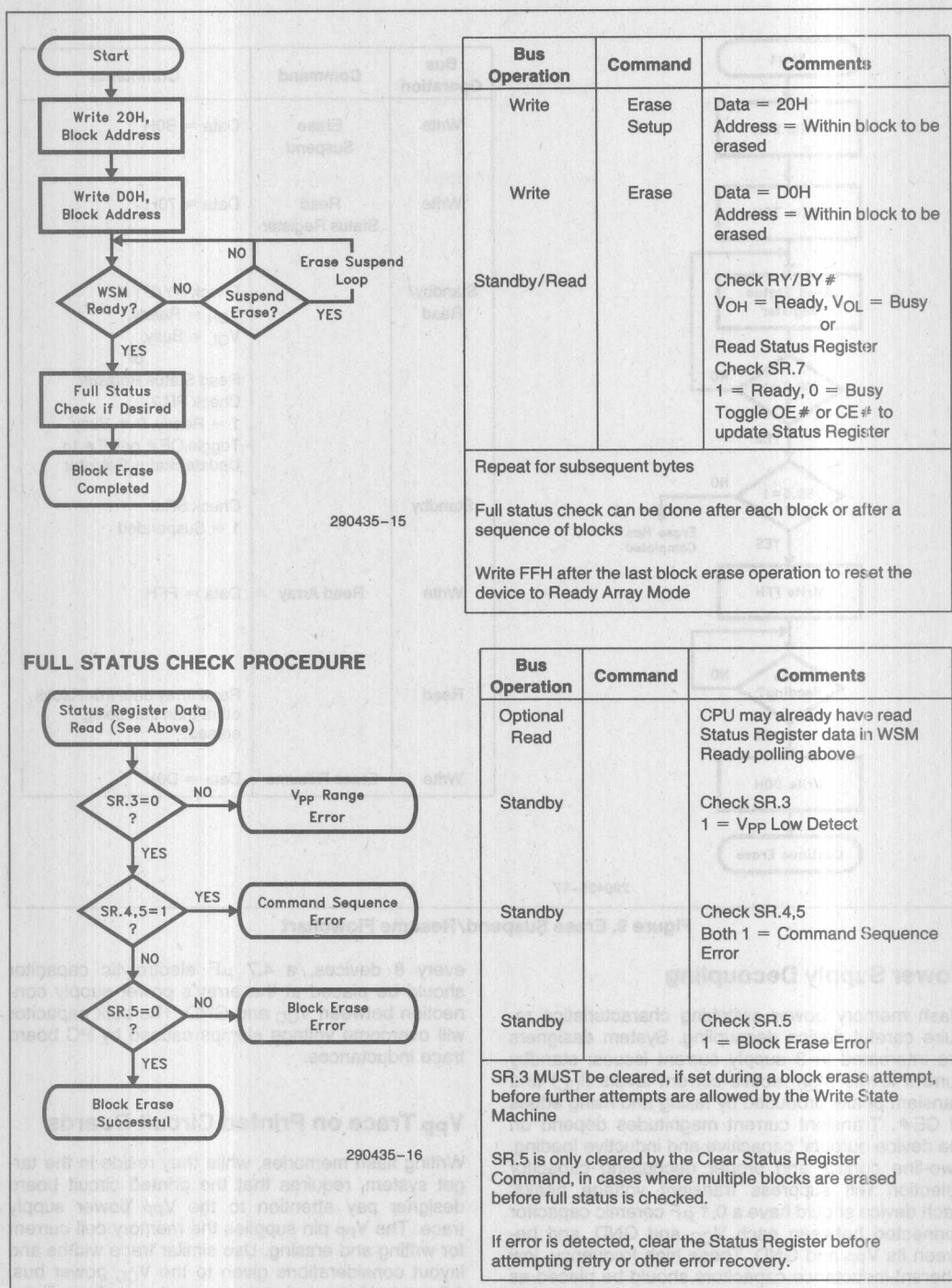


Figure 8. Automated Block Erase Flowchart

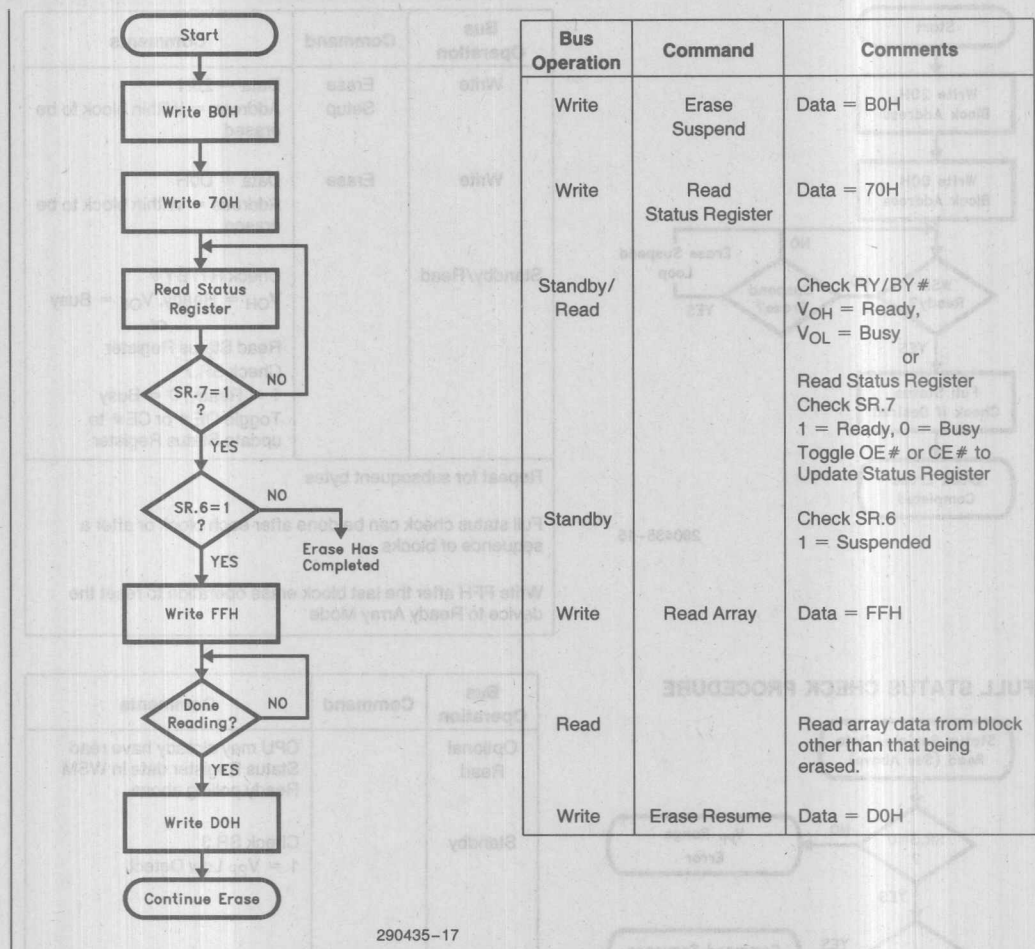


Figure 9. Erase Suspend/Resume Flowchart

Power Supply Decoupling

Flash memory power switching characteristics require careful device decoupling. System designers are interested in 3 supply current issues; standby current levels (I_{SB}), active current levels (I_{CC}) and transient peaks produced by falling and rising edges of CE#. Transient current magnitudes depend on the device outputs' capacitive and inductive loading. Two-line control and proper decoupling capacitor selection will suppress transient voltage peaks. Each device should have a 0.1 μ F ceramic capacitor connected between each V_{CC} and GND, and between its V_{PP} and GND. These high frequency, low inherent-inductance capacitors should be placed as close as possible to package leads. Additionally, for

every 8 devices, a 4.7 μ F electrolytic capacitor should be placed at the array's power supply connection between V_{CC} and GND. The bulk capacitor will overcome voltage slumps caused by PC board trace inductances.

V_{PP} Trace on Printed Circuit Boards

Writing flash memories, while they reside in the target system, requires that the printed circuit board designer pay attention to the V_{PP} power supply trace. The V_{PP} pin supplies the memory cell current for writing and erasing. Use similar trace widths and layout considerations given to the V_{CC} power bus. Adequate V_{PP} supply traces and decoupling will decrease V_{PP} voltage spikes and overshoots.

ABSOLUTE MAXIMUM RATINGS*

Operating Temperature	
During Read	0°C to +70°C(1)
During Block Erase/Byte Write	0°C to +70°C
Temperature Under Bias	-10°C to +80°C
Storage Temperature	-65°C to +125°C
Voltage on Any Pin	
(except V _{CC} and V _{PP})	
with Respect to GND	-2.0V to +7.0V(2)
V _{PP} Program Voltage with	
Respect to GND during	
Block Erase/Byte Write	-2.0V to +14.0V(2, 3)
V _{CC} Supply Voltage	
with Respect to GND	-2.0V to +7.0V(2)
Output Short Circuit Current	100 mA(4)

NOTES:

1. Operating temperature is for commercial product defined by this specification.
2. Minimum DC voltage is -0.5V on input/output pins. During transitions, this level may undershoot to -2.0V for periods <20 ns. Maximum DC voltage on input/output pins is V_{CC} + 0.5V which, during transitions, may overshoot to V_{CC} + 2.0V for periods <20 ns.
3. Maximum DC voltage on V_{PP} may overshoot to +14.0V for periods <20 ns.
4. Output shorted for no more than one second. No more than one output shorted at a time.
5. AC specifications are valid at both voltage ranges. See DC Characteristics for voltage range specific specification.

OPERATING CONDITIONS

Symbol	Parameter	Notes	Min	Max	Unit
T _A	Operating Temperature		0	70	°C
V _{CC}	V _{CC} Supply Voltage	5	3.00	3.60	V
V _{CC}	V _{CC} Supply Voltage	5	4.50	5.50	V

DC CHARACTERISTICS V_{CC} = 3.3V ±0.3V

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Condition
I _{LI}	Input Load Current	1			±0.5	μA	V _{CC} = V _{CC} Max V _{IN} = V _{CC} or GND
I _{LO}	Output Leakage Current	1			±0.5	μA	V _{CC} = V _{CC} Max V _{OUT} = V _{CC} or GND
I _{CCS}	V _{CC} Standby Current	1, 3		20	50	μA	V _{CC} = V _{CC} Max CE# = RP# = V _{IH}
				30	100	μA	V _{CC} = V _{CC} Max CE# = RP# = V _{CC} ±0.2V
I _{CCD}	V _{CC} Deep PowerDown Current	1		0.20	1.0	μA	RP# = GND ±0.2V I _{OUT} (RY/BY#) = 0 mA
I _{CCR}	V _{CC} Read Current	1		5	12	mA	V _{CC} = V _{CC} Max, CE# = GND f = 5 MHz, I _{OUT} = 0 mA CMOS Inputs
				5	12	mA	V _{CC} = V _{CC} Max, CE# = V _{IL} f = 8 MHz, I _{OUT} = 0 mA TTL Inputs

NOTICE: This data sheet contains information on products in the sampling and initial production phases of development. The specifications are subject to change without notice. Verify with your local Intel Sales office that you have the latest data sheet before finalizing a design.

*WARNING: Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.

DC CHARACTERISTICS (Continued)

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Condition
I _{CCW}	V _{CC} Byte Write Current	1		6	18	mA	Byte Write In Progress
I _{CCE}	V _{CC} Block Erase Current	1		6	18	mA	Block Erase In Progress
I _{CCES}	V _{CC} Erase Suspend Current	1, 2		3	6	mA	Block Erase Suspended CE# = V _{IH}
I _{PPS}	V _{PP} Standby Current	1		±1	±15	μA	V _{PP} ≤ V _{CC}
				90	200	μA	V _{PP} > V _{CC}
I _{PPD}	V _{PP} Deep PowerDown Current	1		0.10	5.0	μA	RP# = GND ±0.2V
I _{PPW}	V _{PP} Byte Write Current	1		10	30	mA	V _{PP} = V _{PPH} Byte Write in Progress
I _{PPE}	V _{PP} Block Erase Current	1		10	30	mA	V _{PP} = V _{PPH} Block Erase in Progress
I _{PPES}	V _{PP} Erase Suspend Current	1		90	200	μA	V _{PP} = V _{PPH} Block Erase Suspended
V _{IL}	Input Low Voltage		-0.5		0.6	V	
V _{IH}	Input High Voltage		2.0		V _{CC} + 0.5	V	
V _{OL}	Output Low Voltage	3			0.4	V	V _{CC} = V _{CC} Min I _{OL} = 2 mA
V _{OH}	Output High Voltage	3	2.4			V	V _{CC} = V _{CC} Min I _{OH} = -2 mA
V _{PPL}	V _{PP} during Normal Operations	4	0.0		6.5	V	
V _{PPH}	V _{PP} during Erase/Write Operations		11.4	12.0	12.6	V	
V _{LKO}	V _{CC} Erase/Write Lock Voltage		2.0			V	

CAPACITANCE(5) T_A = 25°C, f = 1 MHz

Symbol	Parameter	Typ	Max	Unit	Condition
C _{IN}	Input Capacitance	6	8	pF	V _{IN} = 0V
C _{OUT}	Output Capacitance	8	12	pF	V _{OUT} = 0V

NOTES:

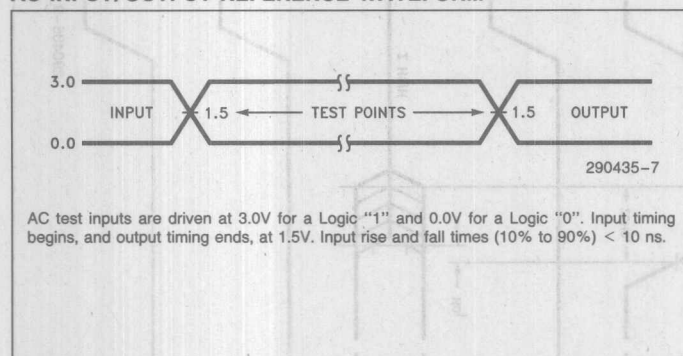
1. All currents are in RMS unless otherwise noted. Typical values at V_{CC} = 3.3V, V_{PP} = 12.0V, T = 25°C. These currents are valid for all product versions (packages and speeds).
2. I_{CCES} is specified with the device deselected. If the 28F008SA-L is read while in Erase Suspend Mode, current draw is the sum of I_{CCES} and I_{CCR}.
3. Includes RY/BY#.
4. Block Erases/Byte Writes are inhibited when V_{PP} = V_{PPL} and not guaranteed in the range between V_{PPH} and V_{PPL}.
5. Sampled, not 100% tested.

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Condition
I _{LI}	Input Load Current	1			±1.0	μA	V _{CC} = V _{CC} Max V _{IN} = V _{CC} or GND
I _{LO}	Output Leakage Current	1			±10	μA	V _{CC} = V _{CC} Max V _{OUT} = V _{CC} or GND
I _{CCS}	V _{CC} Standby Current	1, 3		1.0	2.0	mA	V _{CC} = V _{CC} Max CE# = RP# = V _{IH}
				30	100	μA	V _{CC} = V _{CC} Max CE# = RP# = V _{CC} ± 0.2V
I _{CCD}	V _{CC} Deep PowerDown Current	1		0.20	1.2	μA	RP# = GND ± 0.2V I _{OUT} (RY/BY#) = 0 mA
I _{CCR}	V _{CC} Read Current	1		20	35	mA	V _{CC} = V _{CC} Max, CE# = GND f = 5 MHz, I _{OUT} = 0 mA CMOS Inputs
				25	50	mA	V _{CC} = V _{CC} Max, CE# = V _{IL} f = 5 MHz, I _{OUT} = 0 mA TTL Inputs
I _{CCW}	V _{CC} Byte Write Current	1		10	30	mA	Byte Write In Progress
I _{CCE}	V _{CC} Block Erase Current	1		10	30	mA	Block Erase In Progress
I _{CCES}	V _{CC} Erase Suspend Current	1, 2		5	10	mA	Block Erase Suspended, CE# = V _{IH}
I _{PPS}	V _{PP} Standby Current	1		±1	±15	μA	V _{PP} ≤ V _{CC}
I _{PPD}	V _{PP} Deep PowerDown Current	1		0.10	5.0	μA	RP# = GND ± 0.2V
I _{PPR}	V _{PP} Read Current	1		90	200	μA	V _{PP} > V _{CC}
I _{PPW}	V _{PP} Byte Write Current	1		10	30	mA	V _{PP} = V _{PPH} Byte Write in Progress
I _{PPE}	V _{PP} Block Erase Current	1		10	30	mA	V _{PP} = V _{PPH} Block Erase in Progress
I _{PPES}	V _{PP} Erase Suspend Current	1		90	200	μA	V _{PP} = V _{PPH} Block Erase Suspended
V _{IL}	Input Low Voltage		-0.5		0.8	V	
V _{IH}	Input High Voltage		2.0		V _{CC} + 0.5	V	
V _{OL}	Output Low Voltage	3			0.45	V	V _{CC} = V _{CC} Min I _{OL} = 5.8 mA
V _{OH}	Output High Voltage	3	2.4			V	V _{CC} = V _{CC} Min I _{OH} = -2.5 mA
V _{PPL}	V _{PP} during Normal Operations	4	0.0		6.5	V	
V _{PPH}	V _{PP} during Erase/Write Operations		11.4	12.0	12.6	V	
V _{LKO}	V _{CC} Erase/Write Lock Voltage		2.0			V	

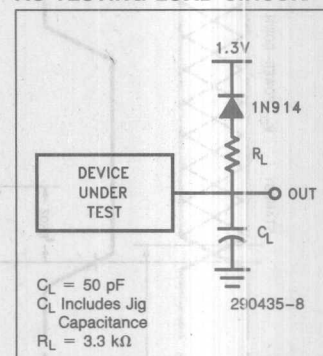
NOTES:

1. All currents are in RMS unless otherwise noted. Typical values at V_{CC} = 5.0V, V_{PP} = 12.0V, T = 25°C. These currents are valid for all product versions (packages and speeds).
2. I_{CCES} is specified with the device deselected. If the 28F008SA-L is read while in Erase Suspend Mode, current draw is the sum of I_{CCES} and I_{CCR}.
3. Includes RY/BY#.
4. Block Erases/Byte Writes are inhibited when V_{PP} = V_{PPL} and not guaranteed in the range between V_{PPH} and V_{PPL}.

AC INPUT/OUTPUT REFERENCE WAVEFORM



AC TESTING LOAD CIRCUIT(2)



AC CHARACTERISTICS—Read-Only Operations(1) $V_{CC} = 3.3V \pm 0.3V, 5.0V \pm 10\%$

Versions			28F008SA-L200		Unit
Symbol	Parameter	Notes	Min	Max	
t_{AVAV}	t_{RC}	Read Cycle Time	200		ns
t_{AVQV}	t_{ACC}	Address to Output Delay		200	ns
t_{ELQV}	t_{CE}	CE # to Output Delay	2	200	ns
t_{PHQV}	t_{PWH}	RP # High to Output Delay		500	ns
t_{GLQV}	t_{OE}	OE # to Output Delay	2	85	ns
t_{ELQX}	t_{LZ}	CE # to Output Low Z	3	0	ns
t_{EHQZ}	t_{HZ}	CE # High to Output High Z	3	55	ns
t_{GLQX}	t_{OLZ}	OE # to Output Low Z	3	0	ns
t_{GHQZ}	t_{DF}	OE # High to Output High Z	3	30	ns
	t_{OH}	Output Hold from Addresses, CE # or OE # Change, Whichever is First	3	0	ns

NOTES:

- See AC Input/Output Reference Waveform for timing measurements.
- OE # may be delayed up to $t_{CE} - t_{OE}$ after the falling edge of CE # without impact on t_{CE} .
- Sampled, not 100% tested.

290435-9

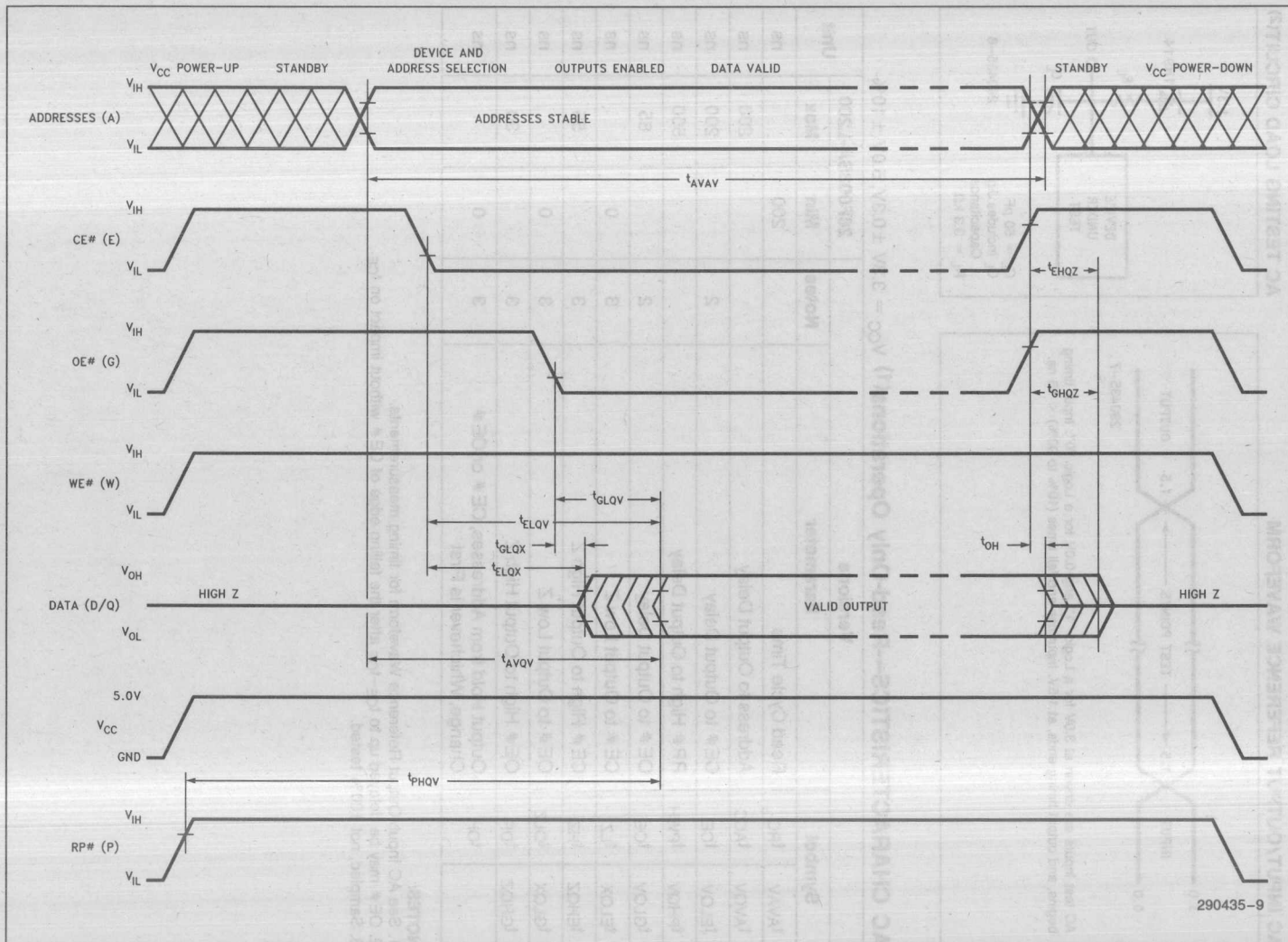


Figure 10. AC Waveform for Read Operations

AC CHARACTERISTICS—Write Operations⁽¹⁾ $V_{CC} = 3.3V \pm 0.3V, 5.0V \pm 10\%$

Versions				28F008SA-L200		Unit
Symbol		Parameter	Notes	Min	Max	
t_{AVAV}	t_{WC}	Write Cycle Time		200		ns
t_{PHWL}	t_{PS}	RP # High Recovery to WE # Going Low	2	1		μs
t_{ELWL}	t_{CS}	CE # Setup to WE # Going Low		20		ns
t_{WLWH}	t_{WP}	WE # Pulse Width		60		ns
t_{VPWH}	t_{VPS}	V_{PP} Setup to WE # Going High	2	100		ns
t_{AVWH}	t_{AS}	Address Setup to WE # Going High	3	60		ns
t_{DVWH}	t_{DS}	Data Setup to WE # Going High	4	60		ns
t_{WHDX}	t_{DH}	Data Hold from WE # High		5		ns
t_{WHAX}	t_{AH}	Address Hold from WE # High		5		ns
t_{WHEH}	t_{CH}	CE # Hold from WE # High		10		ns
t_{WHWL}	t_{WPH}	WE # Pulse Width High		30		ns
t_{WHRL}		WE # High to RY/BY # Going Low			100	ns
t_{WHQV1}		Duration of Byte Write Operation	5, 6	6		μs
t_{WHQV2}		Duration of Block Erase Operation	5, 6	0.3		sec
t_{WHGL}		Write Recovery before Read		0		μs
t_{QVVL}	t_{VPH}	V_{PP} Hold from Valid SRD, RY/BY # High	2, 6	0		ns

NOTES:

1. Read timing characteristics during erase and byte write operations are the same as during read-only operations. Refer to AC Characteristics for Read-Only Operations.
2. Sampled, not 100% tested.
3. Refer to Table 3 for valid A_{IN} for byte write or block erasure.
4. Refer to Table 3 for valid D_{IN} for byte write or block erasure.
5. The on-chip Write State Machine incorporates all byte write and block erase system functions and overhead of standard Intel flash memory, including byte program and verify (byte write) and block precondition, precondition verify, erase and erase verify (block erase).
6. Byte write and block erase durations are measured to completion ($SR.7 = 1, RY/BY \# = V_{OH}$). V_{PP} should be held at V_{PPH} until determination of byte write/block erase success ($SR.3/4/5 = 0$).

BLOCK ERASE AND BYTE WRITE PERFORMANCE $V_{CC} = 3.3V \text{ to } 0.3V, 5.0V \pm 10\%$

Parameter	Notes	28F008SA-L-200			Unit
		Min	Typ(1)	Max	
Block Erase Time	2		2.0	12.5	sec
Block Write Time	2		0.7	2.6	sec

NOTES:

- 25°C, 12.0 Vpp.
- Excludes System-Level Overhead.

1. Read timing characteristics during erase and byte write operations are the same as during read-only operations. Refer to AC Characteristics for Read-Only Operations.

2. Sampled, not 100% tested.

3. Refer to Table 3 for valid A_{16} for byte write or block erase.

4. Refer to Table 3 for valid D_{16} for byte write or block erase.

5. The on-chip Write State Machine incorporates all byte write and block erase system functions and overhead of standard Intel flash memory, including byte program and verify (byte write) and block precondition, precondition verify, erase and erase verify (block erase).

6. Byte write and block erase duration are measured to completion ($BR_1 = 1$, $RYBY_1 = V_{CC}$). V_{PP} should be held at V_{PP} until completion of byte write/block erase success ($CR_1/RY_1 = 0$).

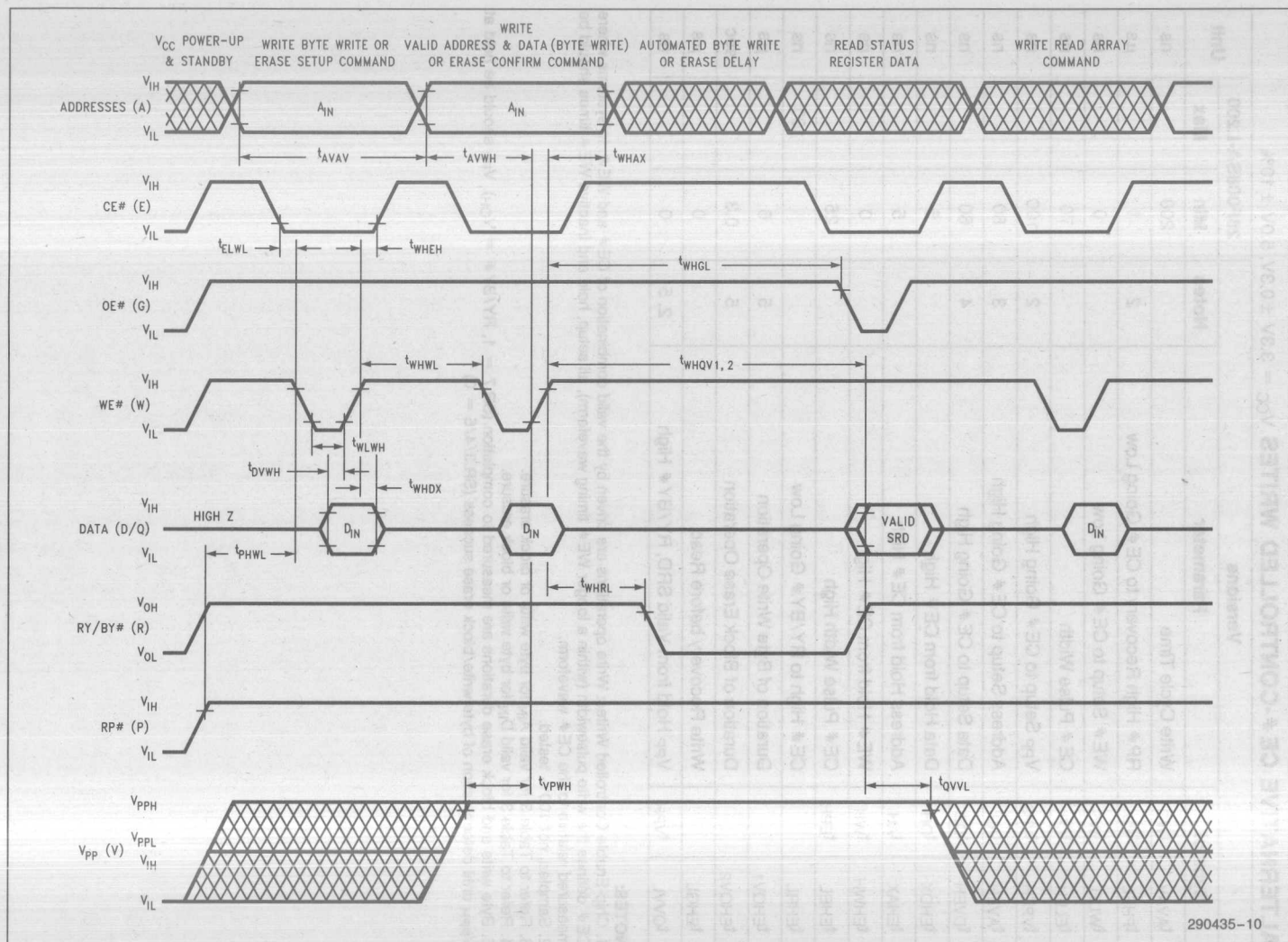


Figure 11. AC Waveform for Write Operations

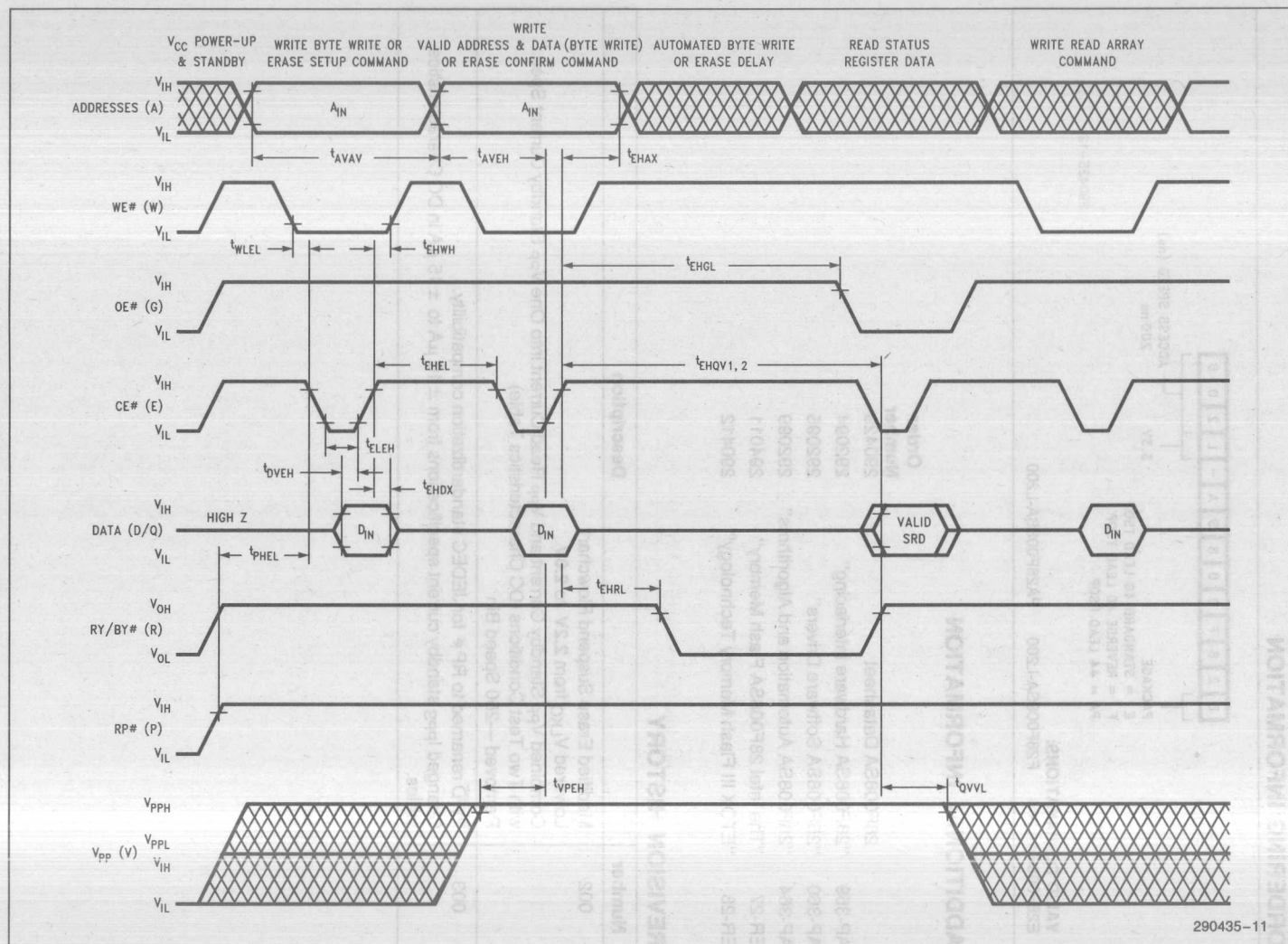
ALTERNATIVE CE #-CONTROLLED WRITES $V_{CC} = 3.3V \pm 0.3V, 5.0V \pm 10\%$

Versions			28F008SA-L200		Unit
Symbol	Parameter	Notes	Min	Max	
t_{AVAV} t_{WC}	Write Cycle Time		200		ns
t_{PHEL} t_{PS}	RP # High Recovery to CE # Going Low	2	1		μs
t_{WLEL} t_{WS}	WE # Setup to CE # Going Low		0		ns
t_{ELEH} t_{CP}	CE # Pulse Width		70		ns
t_{VPEH} t_{VPS}	V _{PP} Setup to CE # Going High	2	100		ns
t_{AVEH} t_{AS}	Address Setup to CE # Going High	3	60		ns
t_{DVEH} t_{DS}	Data Setup to CE # Going High	4	60		ns
t_{EHDH} t_{DH}	Data Hold from CE # High		5		ns
t_{EHAX} t_{AH}	Address Hold from CE # High		5		ns
t_{EHWL} t_{WH}	WE # Hold from CE # High		0		ns
t_{EHEL} t_{EPH}	CE # Pulse Width High		25		ns
t_{EHRL}	CE # High to RY/BY # Going Low			100	ns
t_{EHQV1}	Duration of Byte Write Operation	5	6		μs
t_{EHQV2}	Duration of Block Erase Operation	5	0.3		sec
t_{EHGL}	Write Recovery before Read		0		μs
t_{QVVL} t_{VPH}	V _{PP} Hold from Valid SRD, RY/BY # High	2, 5	0		ns

NOTES:

1. Chip-Enable Controlled Writes: Write operations are driven by the valid combination of CE # and WE #. In systems where CE # defines the write pulsewidth (within a longer WE # timing waveform), all setup, hold and inactive WE # times should be measured relative to the CE # waveform.
2. Sampled, not 100% tested.
3. Refer to Table 3 for valid A_{IN} for byte write or block erasure.
4. Refer to Table 3 for valid D_{IN} for byte write or block erasure.
5. Byte write and block erase durations are measured to completion (SR.7 = 1, RY/BY # = V_{OH}). V_{PP} should be held at V_{PPH} until determination of byte write/block erase success (SR.3/4/5 = 0)

Figure 12. Alternate AC Waveform for Write Operations



290435-11

ORDERING INFORMATION

E	2	8	F	0	0	8	S	A	-	L	2	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---

PACKAGE

E = STANDARD 40 LEAD TSOP

F = REVERSE 40 LEAD TSOP

PA = 44 LEAD PSOP

3.3V

ACCESS SPEED (ns)

200 ns

290435-12

VALID COMBINATIONS:

E28F008SA-L200

F28F008SA-L200

PA28F008SA-L200

290435-12

ADDITIONAL INFORMATION

		Order Number
	28F008SA Datasheet	290429
AP-359	"28F008SA Hardware Interfacing"	292094
AP-360	"25F008SA Software Drivers"	292095
AP-364	"28F008SA Automation and Algorithms"	292099
ER-27	"The Intel 28F008SA Flash Memory"	294011
ER-28	"ETOX III Flash Memory Technology"	290412

REVISION HISTORY

Number	Description
002	Modified Erase Suspend Flowchart Lowered V_{LKO} from 2.2V to 2.0V Combined V_{PP} Standby Current and V_{PP} Read Current into One V_{PP} Standby Current Spec. with Two Test Conditions (DC Characteristics Table) Removed —250 Speed Bin
003	PWD renamed to RP# for JEDEC standardization compatibility. Changed I_{PPS} standby current specifications from $\pm 10 \mu A$ to $\pm 15 \mu A$ in DC Characteristics tables.



AP-359

APPLICATION NOTE

28F008SA Hardware Interfacing

BRIAN DIPERT
MCD MARKETING APPLICATIONS

September 1993

CONTENTS

PAGE

1.0 INTRODUCTION	3-107
2.0 HARDWARE INTERFACING	3-108
2.1 V _{pp} (Byte Write/Block Erase Voltage)	3-108
V _{pp} Generation Circuits	3-109
Controlling V _{pp} to 28F008SA Component(s)	3-109
2.2 RY/BY# (Ready/Busy) Output ..	3-110
2.3 RP# (Reset/Powerdown) Input ..	3-110
Deep Powerdown Mode	3-110
Write Protection	3-111
Reset Control	3-111
2.4 WE# (Write Enable) Input	3-111
2.5 High Density/In ² Layout	3-112
2.6 Power Supply Decoupling	3-112
2.7 High Speed Design Techniques ..	3-113
2.8 Example Bus Interfaces	3-113

CONTENTS

PAGE

ADDITIONAL INFORMATION	3-113
APPENDIX A: Intel386™ SL PI Bus Interface	3-114
APPENDIX B: Intel486™ SX Local CPU Bus Interface	3-115

1.0 INTRODUCTION

The 28F008SA FlashFile™ Memory is a very high performance 8 Mbit (8,388,608 bit) memory, organized as 1 Mbyte (1,048,576 bytes) of 8 bits each. The 28F008SA contains sixteen 64 Kbyte (65,536 byte) blocks, each block separately erasable and capable of 100,000 byte write-block erase cycles. On-chip automation dramatically simplifies software algorithms, and frees the system microprocessor to service higher priority tasks during component data update. An enhanced system interface allows switching the 28F008SA into a deep powerdown mode during periods of inactivity, and gives a hardware indication of the status of the internal Write State Machine. High-speed access time allows minimal wait-state interfacing to microprocessor buses, and advanced packaging provides optimum density/in².

Features of the 28F008SA include:

- High-Density Symmetrically Blocked Architecture:
 - Sixteen 64 Kbyte Blocks
- Extended Cycling Capability
 - 100,000 Block Erase Cycles
 - 1.6 Million Block Erase Cycles per Chip

- Automated Byte Write and Block Erase
 - Command User Interface
 - Status Register
- System Performance Enhancements
 - RY/BY# Status Output
 - Erase Suspend Capability
- Deep Powerdown Mode
 - 0.20 μ A I_{CC} Typical
- Very High Performance Read
 - 85 ns Maximum Access Time
- SRAM-Compatible Write Interface
- Hardware Data Protection Features
 - Erase/Write Lockout during Power Transitions
- Industry Standard Packaging
 - 40 Lead TSOP, 44 Lead PSOP
- ETOX III Nonvolatile Flash Memory Technology
 - 12V Byte Write/Block Erase

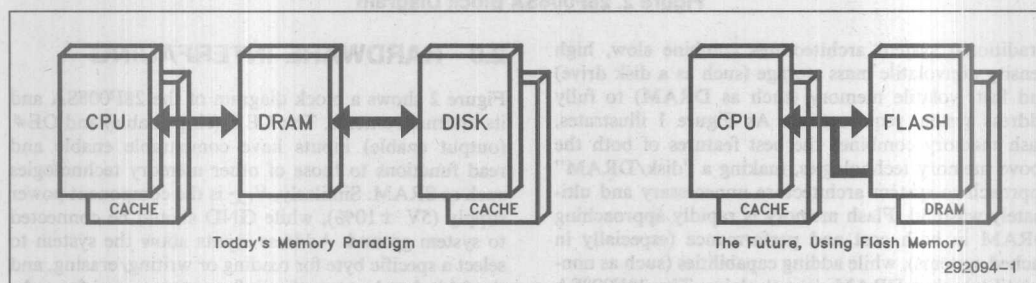


Figure 1. The 28F008SA Revolutionizes the Architecture of Computing

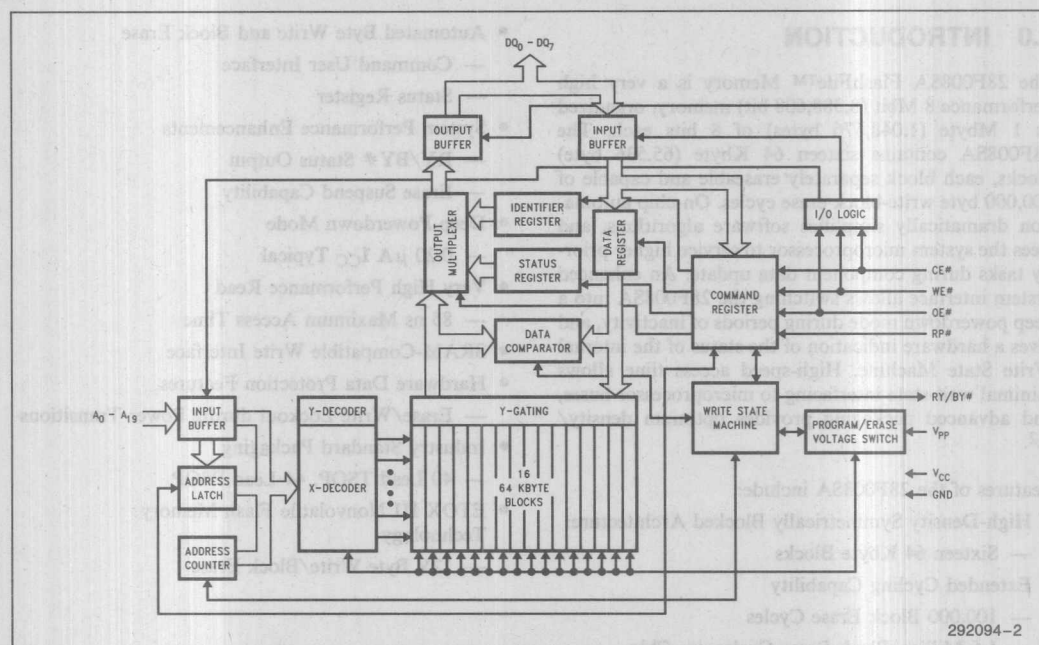


Figure 2. 28F008SA Block Diagram

Traditional system architectures combine slow, high density nonvolatile mass storage (such as a disk drive) and fast, volatile memory (such as DRAM) to fully address system requirements. As Figure 1 illustrates, flash memory combines the best features of both the above memory technologies, making a "disk/DRAM" approach to system architecture unnecessary and ultimately wasteful. Flash memory is rapidly approaching DRAM in both cost and performance (especially in cached systems), while adding capabilities (such as non-volatility), that DRAM cannot claim. The 28F008SA will be the building block memory of choice for emerging computing markets, whether integrated in a memory card or disk drive form factor, or resident on the system motherboard.

This application note discusses hardware interfacing of the 28F008SA flash memory to system designs. The 28F008SA datasheet (order number 290429) is a valuable reference document, providing in-depth device technical specifications, package pinouts and timing waveforms. Additionally, companion application note AP-360, "28F008SA Software Drivers" (order number 292095) provides example ASM-86 and "C" routines for controlling the 28F008SA. AP-364 "28F008SA Automation and Algorithms" discusses in-depth operation of the 28F008SA Write State Machine and internal algorithms, emphasizing how they interface to system software and hardware. AP-360 and AP-364 should be reviewed in conjunction with this application note and the 28F008SA datasheet for a complete understanding of this device.

2.0 HARDWARE INTERFACING

Figure 2 shows a block diagram of the 28F008SA and its internal contents. The CE# (chip enable) and OE# (output enable) inputs have comparable enable and read functions to those of other memory technologies such as SRAM. Similarly, V_{CC} is the component power supply (5V \pm 10%), while GND should be connected to system ground. Address inputs allow the system to select a specific byte for reading or writing/erasing, and the 8-bit data bus transfers information to and from the 28F008SA. The other control lines (WE#, RP#, RY/BY# and V_{pp}) are discussed below.

2.1 V_{PP} (Byte Write/Block Erase Voltage)

The V_{pp} input supplies high voltage to the 28F008SA to enable byte write and block erase. V_{pp} is specified at 12V ± 5% (11.4V–12.6V). Attempting to byte write or block erase the 28F008SA beyond the 5% 12V tolerance is not recommended. V_{pp} above 12.6V can potentially result in device damage, and V_{pp} below 11.4V dramatically lengthens write/erase time and compromises data reliability. The 28F008SA is guaranteed to prevent byte write and block erase attempts with V_{pp} below 6.5V, and in this situation it reports a “low V_{pp} error” through the component Status Register (see AP-360, AP-364 or the 28F008SA datasheet).

V_{pp} Generation Circuits

12V is often already present in systems, used to power the hard drive, display, RS-232 circuitry, flash BIOS update, etc. If it meets the tolerance and current capability requirements of the 28F008SA, such a power supply could be used directly as the 28F008SA update voltage source. However, 12V is sometimes not present or otherwise required, and in such cases, the 28F008SA V_{pp} must be derived from existing voltages and supplies.

Fortunately, flash memory's rapidly increasing popularity has driven ever-improving 12V converter availability in the market. These solutions derive a regulated 12V from a wide range of input voltages, and offer varied levels of integration and current delivery capability. In general, the input for 12V converters should come from the unregulated system power source, particularly in battery-powered systems.

Table 1 lists and briefly describes several 12V generation solutions available at the time this document was published. This is by no means an exhaustive list, and does not reflect any specific recommendation by Intel Corporation. For in-depth information on power supply solutions for flash memory, reference Intel application note AP-357 (order number 292092), available through your local Intel sales office or distributor.

Controlling V_{pp} to 28F008SA Component(s)

Once 12V is available in the system, how is it controlled? One approach is to hard-wire 12V from the supply directly to the V_{pp} inputs of each 28F008SA in the system. The advantage here is in design simplicity and board space savings. The 28F008SA Command User Interface architecture and two-step byte write/block erase command sequences provide protection from unwanted data alteration even with high voltage present on V_{pp}. All 28F008SA functions are disabled with V_{CC} below lockout voltage V_{LKO} (2.2V), or when

RP# is at V_{IL} (see section 2.3). This provides data protection during system powerup, when the minimally-loaded V_{pp} supply often ramps to 12V before V_{CC} (and therefore control inputs to the device) are stable.

For additional data protection, the system designer can choose to make the V_{pp} supply switchable via a GPIO (General Purpose Input/Output) line, enabling 12V to the 28F008SA only during byte write or block erase attempts. A switchable V_{pp} also minimizes power consumption by both the flash memory components and the 12V supply or converter (due to efficiency losses). Many 12V converters integrate an ENABLE input, eliminating external circuitry. If such an input is not available, a low drain-source resistance MOSFET switch such as the Motorola MTD4P05 can be used at the 12V supply output. An example schematic for this switch is shown in Figure 3. The calculations below show that the low drain-source resistance of the MTD4P05 will keep a 12V input within the 5% tolerance required by the 28F008SA.

$$R_{DS} = 0.6\Omega$$

$$I_{pp} = 60 \text{ mA}$$

(worst case, two components being byte written or block erased)

$$\Delta V_{\text{SWITCH DROP}} = (60 \text{ mA} \times 0.6\Omega) = 0.04\text{V}$$

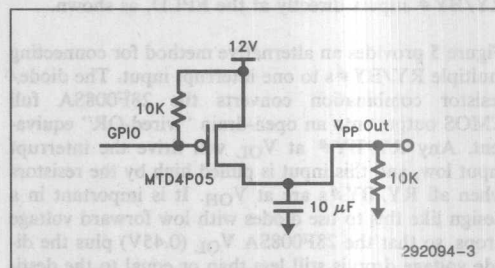


Figure 3. V_{pp} Switch Schematic

Table 1. 12V Conversion Solutions for V_{pp}

Manufacturer	Part Number	Input (V)	Package	Current Output	Total Components Needed	Est. Cost (10K)
Maxim	MAX732	4 to 7.5	16 SOIC	120 mA	9	\$3.93
Linear Technology	LT1110-12	4.5 to 5.5	SO8	120 mA	11	\$4.58
Linear Technology	LT1109-12	4.5 to 5.5	SO8	60 mA	8	\$3.61
Motorola	MC34063A	4.5 to 5.5	SO8	120 mA	15	\$2.25
Maxim	MAX667	12.1 to 16.5	SO8	120 mA	4	\$2.63
Linear Technology	LT1111-12	16 to 30	SO8	120 mA	7	\$3.95
National Semiconductor	LM2940CT-12	13 to 26	TO-220	1A	3	\$1.30

The 28F008SA offers similar automated byte write/block erase capabilities to those first seen in the 28F001BX Bootblock flash memory family, introduced by Intel in May of 1991. It enhances these capabilities via the RY/BY# output, which provides hardware indication of internal Write State Machine (WSM) operation. RY/BY# is a full CMOS output, constantly driven by the 28F008SA and not tristated if the device CE# or OE# inputs are brought to V_{IH} . RY/BY#'s default state after device powerup is V_{OH} . It transitions low to V_{OL} when a byte write or block erase sequence is initiated by system software, and RY/BY#'s rising edge (return to V_{OH}) alerts the system to byte write or block erase completion. RY/BY# also goes to V_{OH} after the 28F008SA is put in Erase Suspend or Deep Powerdown modes.

RY/BY# is intended to interface the 28F008SA to a system microprocessor rising-edge-triggered interrupt input. In a multiple-chip memory array, external EPLD logic or an interrupt controller can be used to combine and prioritize RY/BY#'s into one system interrupt (see Figure 4). The system can then, using a flash memory "activity table" set up in RAM, poll the individual 28F008SA Status Registers to determine which device has returned "ready", or read the RY/BY# inputs directly at the EPLD, as shown.

Figure 5 provides an alternative method for connecting multiple RY/BY#'s to one interrupt input. The diode/resistor combination converts the 28F008SA full CMOS output into an open-drain "wired-OR" equivalent. Any RY/BY# at V_{OL} will drive the interrupt input low, and this input is pulled high by the resistors when all RY/BY#'s are at V_{OH} . It is important in a design like this to use diodes with low forward voltage drops, so that the 28F008SA V_{OL} (0.45V) plus the diode voltage drop is still less than or equal to the destination input V_{IH} (0.8V). For the schematic shown in Figure 5, the equation is:

$$V_{OL} + V_{DIODE} = 0.45 V_{MAX} + 0.3V = 0.75V \leq 0.8V$$

Note that should the system connect RY/BY# to an interrupt, disable that interrupt prior to suspending erase, as RY/BY# will transition to V_{OH} when the device is suspended.

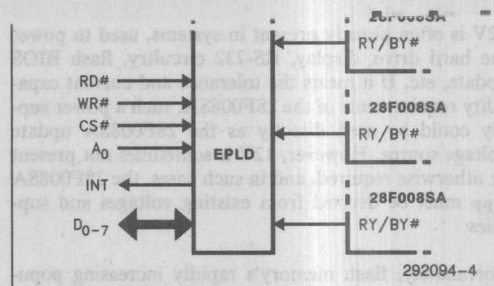


Figure 4. EPLD-Based RY/BY# Implementation

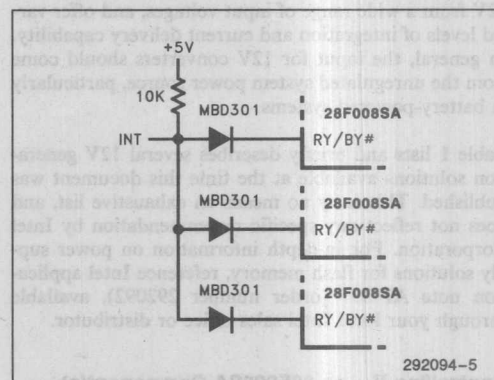


Figure 5. "Wired-OR" RY/BY# Implementation

2.3 RP# (Reset/Powerdown) Input

Deep Powerdown Mode

The RP# input, when driven to V_{IL} by the system, switches the 28F008SA into a deep powerdown mode with negligible power consumption. This feature integrates the V_{CC} power FET often used with low power designs. Power consumption thru V_{CC} is typically 1 μ W in deep powerdown mode. RP#-low deselects the memory, places output drivers for D_{0-7} in a high-impedance state and turns off a majority of internal circuits. RY/BY# is driven to V_{OH} while in deep powerdown mode. Depending on the flexibility desired, system designers can choose to put either the entire flash device array into deep powerdown mode, or any individual components via selective input control. The 28F008SA requires a "wakeup" time after RP# returns to V_{IH} before it can be successfully written (t_{PHWL}) or outputs are valid to read attempts (t_{PHQV}).

Since $RP\# = V_{IL}$ deselects the 28F008SA, this input can be used not only as a means of entering deep powerdown mode but also as an active-high "chip enable" to block spurious writes during system power transitions. Figure 6 shows one possible $RP\#$ implementation, controlled by a GPIO line for power management and by a system POWER GOOD for power sequencing protection. In this design, the 5V monitoring circuit begins functioning at $V_{CC} = 1V$, and will enable the device only after V_{CC} transitions above 4.6V (and system control signals are therefore stable). As V_{CC} drops below 4.6V during system powerdown, $RP\#$ protection is again activated.

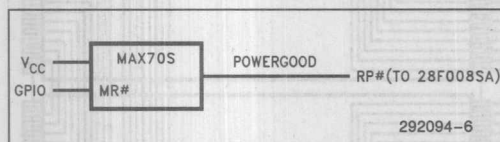


Figure 6. $RP\#$ Gating

Reset Control

$RP\#$ at V_{IL} resets all internal automation within the 28F008SA as part of the deep powerdown process. Upon exit from deep powerdown, the 28F008SA is reset to Read Array mode. This functionality is ideal when the 28F008SA is the boot memory for the system. $RP\#$ active transitions reset the Write Status Machine if system reset occurs during flash memory program or erase, and allow successful CPU reboot.

2.4 WE# (Write Enable) Input

When flash memory is written, the result can range from a 28F008SA that is placed in "read intelligent identifier" or "read Status Register" modes to alteration of nonvolatile flash memory contents. System hardware can prevent spurious writes to flash memory by application software or an operating system by gating the system $WE\#$ to flash memory components to enable writes only when desired.

Figure 7 shows a simple design that gates $WE\#$ with a GPIO line, enabling writes to the 28F008SA only when the GPIO is a "0". The GPIO is initialized to "1" on system powerup and the BIOS, a dedicated update software routine, a special keyboard sequence, switch on the back of the system or jumper on the system motherboard can then control the GPIO. This circuit ensures that flash memory contents are as permanent as "ROM" unless alteration is specifically desired.

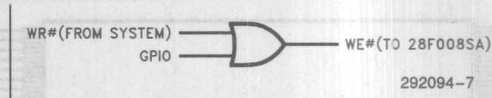


Figure 7. $WE\#$ Gating

2.5 High Density/ ln^2 Layout

Figure 8 shows an 8 Mbyte flash memory array using TSOP (Thin Small Outline)-packaged 28F008SAs in standard (E) and reverse (F) configurations. A layout like this is used in Intel's Series 2 flash memory cards (in densities to 20 Mbytes) and provides optimum array density for available board space.

Address and data lines are connected to all components in parallel. $OE\#$ and $WE\#$ are similarly connected. Section 2.7 of this document discusses alternate methods of implementing these signals for highest speed reads and writes in large memory arrays.

Component $RY/BY\#$ s are shown as not connected in Figure 8. They can be left unused, in which case the system software will substitute polling of component Status Registers for hardware interrupt, or $RY/BY\#$ s can be implemented as described in section 2.2.

$CE\#$ s are also not connected, intended to be individually driven by system chip enable decoding logic. This provides capability to read from and write to the array on a byte-by-byte basis. In a x16-only system, upper and lower byte 28F008SAs can have their $CE\#$ s bused together if desired.

Finally, V_{CC} , V_{pp} and $RP\#$ are connected in parallel to all components. Section 2.6 discusses bypass capacitor filtering of supply voltage inputs, while section 2.3 provides uses for $RP\#$. If desired, individual component, component pair, etc. selective powerdown control can be substituted for the global control shown in Figure 8.

In space-constrained designs, a multiple-layer partial "serpentine" trace layout at the edges of the 28F008SA array may be implemented, with a full serpentine layout within the array as in Figure 8.

3

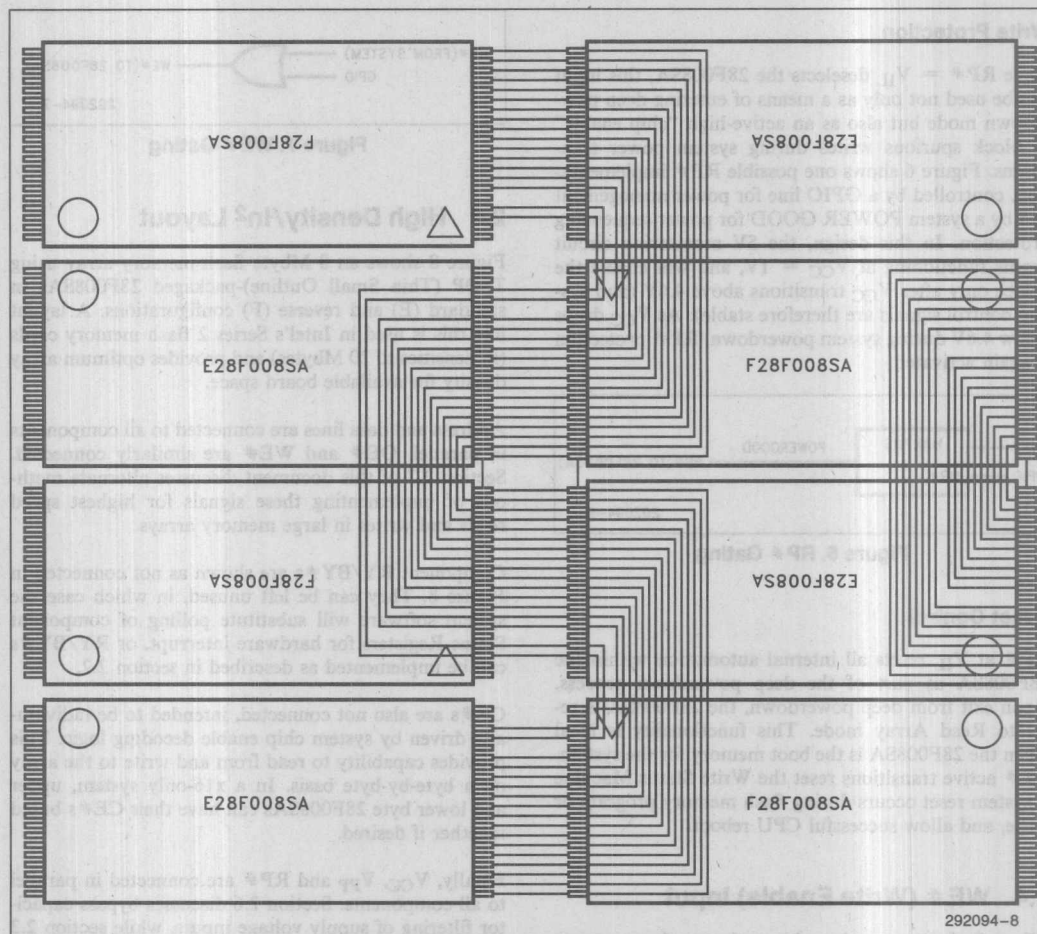


Figure 8. TSOP Serpentine Layout

2.6 Power Supply Decoupling

Both the V_{CC} and V_{pp} inputs to each 28F008SA should be decoupled at the package leads to provide noise immunity and supply current for transient current spikes during read, byte write and block erase. Additional bulk capacitance for groups of flash memories overcomes voltage slump caused by PC board trace inductances. Calculations for individual component and bulk capacitors (one per 8 devices) are shown below.

Basic Equation:

$$I = C \, dv/dt$$

Assumptions:

$$I = 35 \text{ mA per device } (V_{CC}), \text{ therefore}$$

$$I = 17.5 \text{ mA per device input } (V_{CC})$$

$$I = 30 \text{ mA per device } (V_{pp})$$

$$dv = 0.1V \text{ (0.2V peak-peak)}$$

$$dt = 20 \text{ ns}$$

Per-Component-Input Decoupling Capacitor (V_{CC}):

$$C = I \, dt/dv = (17.5 \text{ mA} \times 20 \text{ ns})/0.1V = 3.5 \text{ nF}$$

$$4x \text{ margin} = 4 \times 3.5 \text{ nF} = 14 \text{ nF}$$

$$\text{Standard Equivalent} = 0.01 \, \mu\text{F}$$

NOTE:

Calculations above assume that each 28F008SA is driving CMOS inputs (with corresponding high impedance and negligible input current requirements). If 28F008SA outputs are driving non-CMOS inputs, larger per-component capacitance may be needed to supply current while outputs are switching.

Bulk Capacitor (V_{CC}):

$$C = 10 \times (\text{Total of Decoupling Capacitors})$$

$$\text{Bulk Capacitor (4 Mbyte array)} = 10 \times (8 \times 0.01 \mu\text{F}) = 0.8 \mu\text{F}$$

$$\text{Standard Equivalent} = 1 \mu\text{F}$$

Per-Component Decoupling Capacitor (V_{PP}):

$$C = I \, dt/dv = (30 \text{ mA} \times 20 \text{ ns})/0.1\text{V} = 6 \text{ nF}$$

$$4x \text{ margin} = 4 \times 6 \text{ nF} = 24 \text{ nF}$$

$$\text{Standard Equivalent} = 0.033 \mu\text{F}$$

2.7 High Speed Design Techniques

The 28F008SA's fast read access and command write specifications make it a natural choice for high performance memory arrays. The following tips will optimize the memory interface for optimum read/write speed. The common recommendation in all instances centers around minimizing fanout and capacitive bus loading to allow highest switching speed, lowest rise and fall times, and therefore greatest performance.

ADDITIONAL INFORMATION

	28F008SA Datasheet
	28F008SA-L Datasheet
AP-357	"Power Supply Solutions for Flash Memory"
AP-360	"28F008SA Software Drivers"
AP-364	"28F008SA Automation and Algorithms"
ER-27	"The Intel 28F008SA Flash Memory"
ER-28	"ETOX-III Flash Memory Technology"

- Minimize address bus loading from the microprocessor to the memory array. Multiple address latches feeding subsets of the array speed address input to each 28F008SA and CE# decoding by external logic.
- Similarly, drive the memory array with multiple OE#s and WE#s. Most EPLD and discrete logic timing is specified at a 30 pF load, which equates to driving 4 28F008SA inputs at maximum input capacitance. Anything more than this may severely impact the logic's propagation delay.
- Finally, remember that each 28F008SA, when read, drives not only the system microprocessor or transceiver but also any other flash memory components connected to the common data bus. Each 28F008SA data output is specified at 12 pF, and the 28F008SA read timings are tested at either 30 pF or 100 pF of loading, depending on the chosen speed bin.

For large flash arrays where sequential data can be distributed on many devices, hardware interleaving provides additional performance.

2.8 Example Bus Interfaces

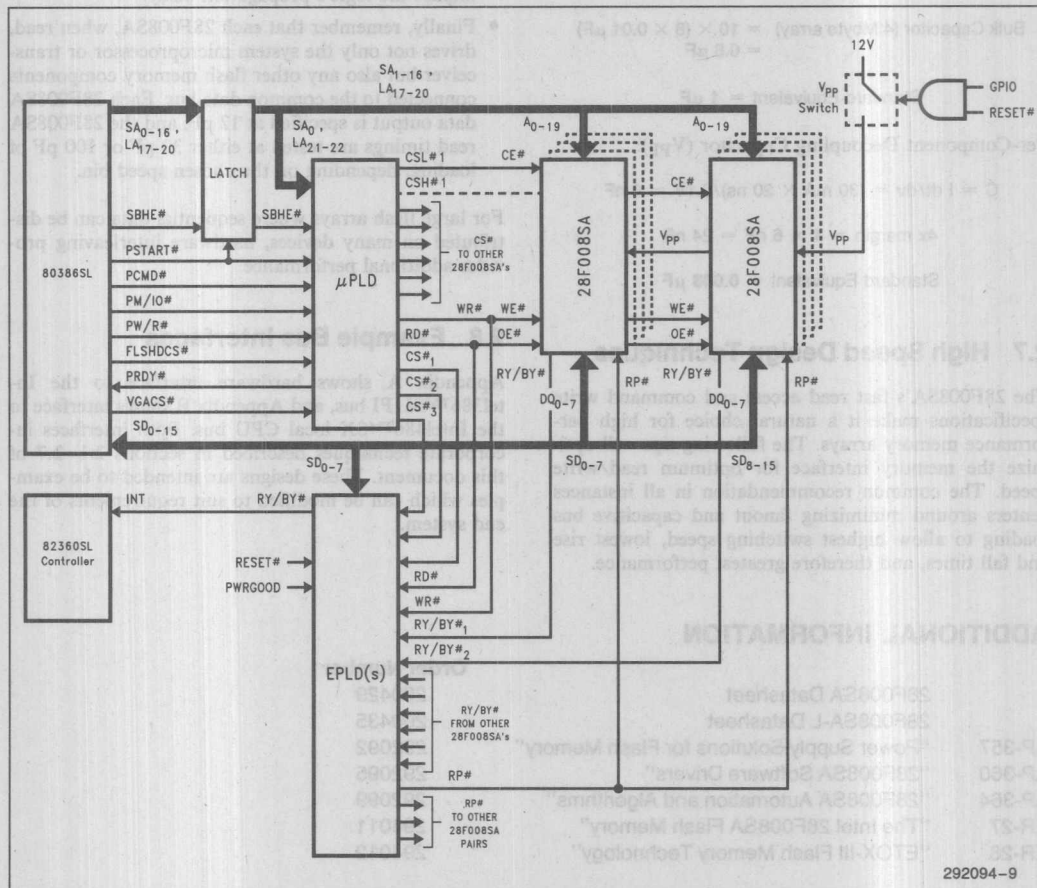
Appendix A shows hardware interface to the Intel386TMSL PI bus, and Appendix B shows interface to the Intel486TMSX local CPU bus. Both interfaces incorporate techniques described in sections 2.1–2.7 of this document. These designs are intended to be examples which can be modified to suit requirements of the end system.

Order Number

290429
290435
292092
292095
292099
294011
294012

APPENDIX A

Intel386™SL PI BUS INTERFACE

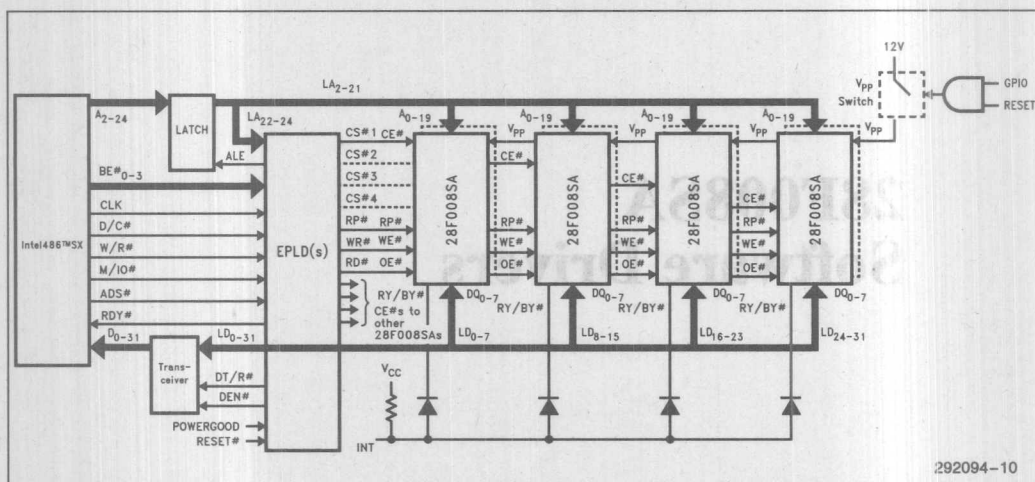


NOTE:

The DRAM interface is not shown, for graphic simplicity.

APPENDIX B

Intel486™ SX LOCAL CPU BUS INTERFACE



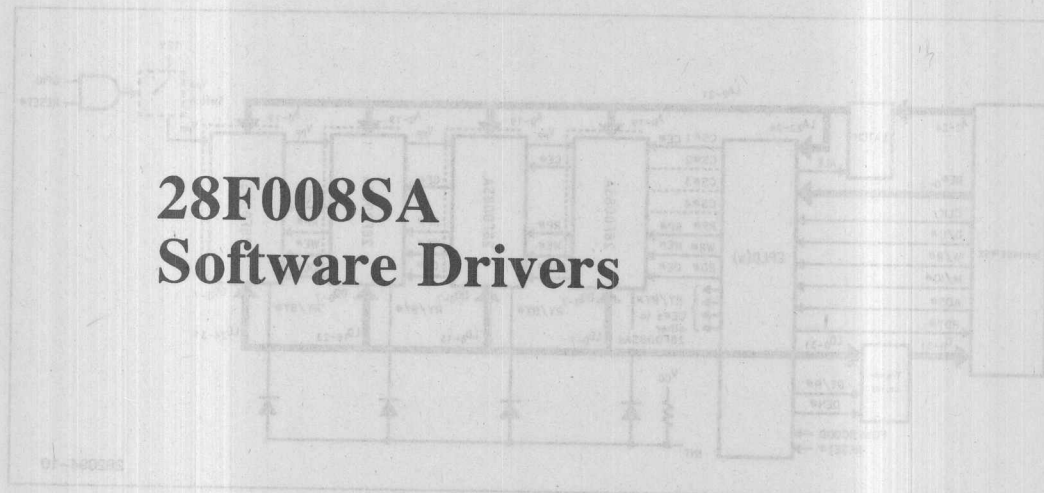
NOTE:
The DRAM interface is not shown, for graphic simplicity.

REVISION HISTORY

Number	Description
-003	Renamed PWD# as RP# to match JEDEC conventions. Updated Figure 6 Added Reset Control discussion for RP# (Reset/Powerdown) Input.

APPLICATION NOTE

28F008SA Software Drivers



NOTE
The DRAM interface is not shown, for graphic simplicity.

REVISION HISTORY

Revision	Description
003	Renamed PWD# as RP# to match JEDEC conventions. Updated Figure 6. Added Reset Control discussion for RP# (Reset) Command input.

BRIAN DIPERT
MCD MARKETING APPLICATIONS

September 1993

28F008SA Software Drivers

CONTENTS	PAGE
1.0 INTRODUCTION	3-118
2.0 ASM86 ASSEMBLY DRIVERS	3-119
3.0 "C" DRIVERS	3-123

The internal automation of the 28F008SA makes software driving logic unnecessary and results in platform-independent code. This software is designed to be executed in any type of memory and with all processors and microprocessors. With ASM-86 assembly code, "C" code can be used with many microprocessors and embedded processors. This code provides the smallest code "kernel" for fast

CONTENTS	PAGE
ADDITIONAL INFORMATION	3-137

This application note provides examples of how to use the 28F008SA in a system. It includes a block diagram of the 28F008SA and a table of its features. The note also provides a detailed description of the 28F008SA's internal architecture and its operation. The note is intended for use by system designers who are interested in using the 28F008SA in their systems. The note is organized into three main sections: Introduction, Assembly Drivers, and "C" Drivers. The Introduction section provides a general overview of the 28F008SA and its features. The Assembly Drivers section provides detailed information about the 28F008SA's internal architecture and its operation. The "C" Drivers section provides detailed information about the 28F008SA's internal architecture and its operation.

This application note provides example software code for byte writing, block erasing and otherwise controlling Intel's 28F008SA 8 Mbit symmetrically blocked FlashFile™ Memory family. Two programming languages are provided; high-level "C" for multi-platform support, and ASM-86 assembly. In many cases, the driver routines can be inserted "as is" into the main body of code being developed by the system software engineer. The text accompanying each routine describes the existing code and suggests area for possible alteration to fit specific applications. These explanations, along with in-line commenting, minimize driver modification efforts.

28F008SA-L are valuable reference documents. Data-sheets should be reviewed in conjunction with this application note for a complete understanding of the devices. AP-359, "28F008SA Hardware Interfacing" is the hardware-oriented application note equivalent for these devices and can also be referenced. AP-364 "28F008SA Automation and Algorithms", another useful reference, discusses the details of Write State machine automation.

The internal automation of the 28F008SA makes software timing loops unnecessary and results in platform-independent code. This software is designed to be executed in any type of memory and with all processor clock rates. "C" code can be used with many microprocessors and microcontrollers, while ASM-86 assembly code provides the smallest code "kernel" for Intel microprocessors and embedded processors.

2.0 ASM-86 DRIVERS

```

; Copyright Intel Corporation, 1992
; Brian Dipert, Intel Corporation, February 8, 1992, Revision 1.0

; Revision History: Rev 1.0

; The following code controls byte write of data to a single 28F008SA (x8 write)
; DS:[SI] points to the data to be written, ES:[DI] is the location to be written
; In protected mode operation, DS and ES reference a descriptor
; Register AX is modified by this procedure
WRITE_SETUP EQU 40H
READ_ID EQU 90H
INTEL_ID EQU 89H
DEVICE_ID EQU 0A2H
DEVICE_ID2 EQU 0A1H
READY EQU 80H
W_ERR_FLAG EQU 10H
VPP_FLAG EQU 08H

; Insert code here to ramp Vpp and disable component RP# input. If a string of bytes is
; to be written at one time, Vpp ramp to 12V and ID check need only occur once.
; before the first byte is written
MOV AX, "Address 0 for target 28F008SA-segment"
; Initialize pointer to 28F008SA address 0
MOV ES, AX
MOV DI, "Address 0 for target 28F008SA-offset"
MOV BYTE PTR ES:[DI], READ_ID ; Write Intelligent Identifier command
CMP BYTE PTR ES:[DI], INTEL_ID ; Does manufacturer ID read correctly?
JNZ W_BYT_ID_ERR
MOV DI, "Address 1 for target 28F008SA-offset"
; Initialize pointer to 28F008SA address 1
CMP BYTE PTR ES:[DI], DEVICE_ID ; Does device ID read correctly?
JZ W_BYT_ID_PASS
CMP BYTE PTR ES:[DI], DEVICE_ID2
JNZ W_BYT_ID_ERR

W_BYT_ID_PASS:
MOV AX, "Byte write destination address-segment"
; Initialize pointer to byte write dest. address
MOV ES, AX
MOV DI, "Byte write destination address-offset"
MOV BYTE PTR ES:[DI], WRITE_SETUP ; Write byte write setup command
MOV AL, DS:[SI] ; Load AL with data to write
MOV ES:[DI], AL ; Write to device

W_BYT_LOOP:
TEST BYTE PTR ES:[DI], READY ; Read 28F008SA Status Register
JZ W_BYT_LOOP ; Loop until bit 7 = 1

TEST BYTE PTR ES:[DI], (W_ERR_FLAG OR VPP_FLAG)
JZ W_BYT_CONT ; Success!

TEST BYTE PTR ES:[DI], W_ERR_FLAG ; Check Status Register bit 4
JNZ W_BYT_ERR ; Jump if = 1, Byte Write Error

TEST ES:[DI], VPP_FLAG ; Check Status Register bit 3
JNZ W_BYT_VPP ; Jump if = 1, Vpp Low Error

W_BYT_ID_ERR:
; Insert code to service improper device ID read error here.
; Is 28F008SA RP# input disabled? Is Vcc applied to the 28F008SA?

W_BYT_ERR:
; Insert code to service byte write error here

W_BYT_VPP:
; Insert code to service byte write Vpp low error here

W_BYT_CONT:
; Code continues from this point....

```

This routine writes a byte of data to a single 28F008SA. Note the use of BYTE PTR notation to force x8 accesses. If a string of bytes is to be written at one time, the Vpp ramp up, RP# disable and device ID checks need only be done before the first byte write attempt. Additionally, when writing multiple bytes at once, examination of bits other than bit 7 (WSM Status) need only occur after the last byte write has completed. The Status Register retains any error bits until the Clear Status Register command is written.

```

; The following code controls byte write of data to a pair of 28F008SAs (x16 write)
; DS:[SI] points to the data to be written, ES:[DI] is the location to be written
; In protected mode operation, DS and ES reference a descriptor
; Register AX is modified by this procedure
WRITE_SETUP EQU 40H
READ_ID EQU 90H
INTEL_ID EQU 89H
DEVICE_ID EQU 0A2H
DEVICE_ID2 EQU 0A1H
READY EQU 80H
W_ERR_FLAG EQU 10H
VPP_FLAG EQU 08H
;
; Insert code here to ramp Vpp and disable component RP# inputs. If a string of words is
; to be written at one time, Vpp ramp to 12V and ID check need only occur once,
; before the first word is written
;
MOV AX, "Address 0 for target 28F008SA-segment"
; Initialize pointer to 28F008SA address 0
MOV ES, AX
MOV DI, "Address 0 for target 28F008SA-offset"
MOV ES:[DI], ((READ_ID SHL 8) OR READ_ID)
; Write Intelligent Identifier command
CMP ES:[DI], ((INTEL_ID SHL 8) OR INTEL_ID)
; Does manufacturer ID read correctly?
JNZ W_WRD_ID_ERR
MOV DI, "Address 1 for target 28F008SA-offset"
; Initialize pointer to 28F008SA address 1
CMP ES:[DI], ((DEVICE_ID SHL 8) OR DEVICE_ID)
; Does device ID read correctly?
JZ W_WRD_ID_PASS
CMP ES:[DI], ((DEVICE_ID2 SHL 8) OR DEVICE_ID2)
JNZ W_WRD_ID_ERR
W_WRD_ID_PASS:
MOV AX, "Byte write destination address-segment"
; Initialize pointer to byte write dest. address
MOV ES, AX
MOV DI, "Byte write destination address-offset"
MOV ES:[DI], ((WRITE_SETUP SHL 8) OR WRITE_SETUP)
; Write byte write setup command
MOV AX, DS:[SI]
; Load AX with data to write
MOV ES:[DI], AX
; Write to devices
W_WRD_LOOP:
TEST ES:[DI], ((READY SHL 8) OR READY)
; Read 28F008SA Status Registers
JZ W_WRD_LOOP
; Loop until bit 7 = 1
TEST ES:[DI], (((W_ERR_FLAG OR VPP_FLAG) SHL 8) OR (W_ERR_FLAG OR VPP_FLAG))
JZ W_WRD_CONT
; Success!
MOV AX, ES:[DI]
; Load Status Register data into AX
TEST AL, W_ERR_FLAG
; Check Status Register bit 4 (low byte)
JNZ W_WRD_ERR
TEST AH, W_ERR_FLAG
; Check Status Register bit 4 (high byte)
JNZ W_WRD_ERR
TEST AL, VPP_FLAG
; Check Status Register bit 3 (low byte)
JNZ W_WRD_VPP
TEST AH, VPP_FLAG
; Check Status Register bit 3 (high byte)
JNZ W_WRD_VPP
W_WRD_ID_ERR:
;
; Insert code to service improper device ID read error here.
; Are 28F008SA RP# inputs disabled? Is Vcc applied to the 28F008SAs?
W_WRD_ERR:
;
; Insert code to service byte write error here
W_WRD_VPP:
;
; Insert code to service byte write Vpp low error here
W_WRD_CONT:
;
; Code continues from this point....

```

This routine writes a word of data to a pair of 28F008SAs. Note that all constants have been "OR'd" for parallel read/write of two devices at once. If a string of words is to be written at one time, the Vpp ramp up, RP# disable and device ID checks need only be done before the first word write attempt. Additionally, when writing multiple words at once, examination of bits other than bit 7 (WSM Status) need only occur after the last word write has completed. The Status Register retains any error bits until the Clear Status Register command is written.

```

; The following code controls block erase of a single 28F008SA (x8 block erase)
; ES:[DI] points to the block to be erased
; In protected mode operation, ES references a descriptor
; Register AX is modified by this procedure
ERASE_SETUP EQU 20H
ERASE_CONFIRM EQU 0D0H
READ_ID EQU 90H
INTEL_ID EQU 89H
DEVICE_ID EQU 0A2H
DEVICE_ID2 EQU 0A1H
READY EQU 80H
E_ERR_FLAG EQU 20H
E_CMD_FLAG EQU 30H
VPP_FLAG EQU 08H

; Insert code here to ramp Vpp and disable component RP# input. If a string of blocks is
; to be erased at one time, Vpp ramp to 12V and ID check need only occur once,
; before the first block is erased
MOV AX, "Address 0 for target 28F008SA-segment"
; Initialize pointer to 28F008SA address 0
MOV ES, AX
MOV DI, "Address 0 for target 28F008SA-offset"
MOV BYTE PTR ES:[DI], READ_ID ; Write Intelligent Identifier command
CMP BYTE PTR ES:[DI], INTEL_ID ; Does manufacturer ID read correctly?
JNZ E_BYT_ID_ERR
MOV DI, "Address 1 for target 28F008SA-offset"
; Initialize pointer to 28F008SA address 1
CMP BYTE PTR ES:[DI], DEVICE_ID ; Does device ID read correctly?
JZ E_BYT_ID_PASS
CMP BYTE PTR ES:[DI], DEVICE_ID2
JNZ E_BYT_ID_ERR

E_BYT_ID_PASS:
MOV AX, "Block erase destination address-segment"
; Initialize pointer to block erase dest.address
MOV ES, AX
MOV DI, "Block erase destination address-offset"
MOV BYTE PTR ES:[DI], ERASE_SETUP ; Write block erase setup command
MOV BYTE PTR ES:[DI], ERASE_CONFIRM ; Write block erase confirm command

E_BYT_LOOP:
TEST BYTE PTR ES:[DI], READY ; Read 28F008SA Status Register
JZ E_BYT_LOOP ; Loop until bit 7 = 1

TEST BYTE PTR ES:[DI], (E_CMD_FLAG OR VPP_FLAG)
JZ E_BYT_CONT ; Success!

TEST BYTE PTR ES:[DI], E_CMD_FLAG ; Check Status Register bits 4 and 5
JNZ E_BYT_CMD_ERR ; Jump if = 1

TEST BYTE PTR ES:[DI], E_ERR_FLAG ; Check Status Register bit 5
JNZ E_BYT_ERR ; Jump if = 1

TEST BYTE PTR ES:[DI], VPP_FLAG ; Check Status Register bit 3
JNZ E_BYT_VPP ; Jump if = 1

E_BYT_ID_ERR:
; Insert code to service improper device ID read error here.
; Is 28F008SA RP# input disabled? Is Vcc applied to the 28F008SA?

E_BYT_CMD_ERR:
; Insert code to service block erase command sequence error here
; (setup followed by a command other than confirm)

E_BYT_ERR:
; Insert code to service block erase error here

E_BYT_VPP:
; Insert code to service block erase Vpp low error here

E_BYT_CONT:
; Code continues from this point.....

```

This routine erases a block of a single 28F008SA. Note the use of BYTE PTR notation to force x8 accesses. If a string of blocks is to be erased at one time, the Vpp ramp up, RP# disable and device ID checks need only be down before the first block erase attempt. Additionally, when erasing multiple blocks at once, examination of bits other than bit 7 (WSM Status) need only occur after the last block erase has completed. The Status Register retains any error bits until the Clear Status Register command is written.

3

```

; The following code controls block erase of a pair of 28F008SAs (x16 block erase)
; ES:[DI] points to the blocks to be erased
; In protected mode operation, ES references a descriptor
; Register AX is modified by this procedure
ERASE_SETUP EQU 20H
ERASE_CONFIRM EQU 0D0H
READ_ID EQU 90H
INTEL_ID EQU 89H
DEVICE_ID EQU 0A2H
DEVICE_ID2 EQU 0A1H
READY EQU 80H
E_ERR_FLAG EQU 20H
E_CMD_FLAG EQU 30H
VPP_FLAG EQU 08H

; Insert code here to ramp Vpp and disable component RP# inputs. If a string of blocks is
; to be erased at one time, Vpp ramp to 12V and ID check need only occur once,
; before the first block pair is erased
MOV AX, "Address 0 for target 28F008SA-segment" ; Initialize pointer to 28F008SA address 0
MOV ES, AX
MOV DI, "Address 0 for target 28F008SA-offset"
MOV ES:[DI], ((READ_ID SHL 8) OR READ_ID) ; Write Intelligent Identifier command
CMP ES:[DI], ((INTEL_ID SHL 8) OR INTEL_ID) ; Does manufacturer ID read correctly?
JNZ E_WRD_ID_ERR
MOV DI, "Address 1 for target 28F008SA-offset" ; Initialize pointer to 28F008SA address 1
CMP ES:[DI], ((DEVICE_ID SHL 8) OR DEVICE_ID) ; Does device ID read correctly?
JZ E_WRD_ID_PASS
CMP ES:[DI], ((DEVICE_ID2 SHL 8) OR DEVICE_ID2)
JNZ E_WRD_ID_ERR

E_WRD_ID_PASS:
MOV AX, "Block erase destination address-segment" ; Initialize pointer to block erase dest. address
MOV ES, AX
MOV DI, "Block erase destination address-offset"
MOV ES:[DI], ((ERASE_SETUP SHL 8) OR ERASE_SETUP) ; Write block erase setup command
MOV ES:[DI], ((ERASE_CONFIRM SHL 8) OR ERASE_CONFIRM) ; Write block erase confirm command

E_WRD_LOOP:
TEST ES:[DI], ((READY SHL 8) OR READY) ; Read 28F008SA Status Registers
JZ E_WRD_LOOP ; Loop until bit 7 = 1
TEST ES:[DI], (((E_CMD_FLAG OR VPP_FLAG) SHL 8) OR (E_CMD_FLAG OR VPP_FLAG))
JZ E_WRD_CONT ; Success!

MOV AX, ES:[DI] ; Load Status Register data into AX
TEST AL, E_CMD_FLAG ; Check Status Reg bits 4 and 5 (low byte)
JNZ E_WRD_CMD_ERR ; Jump if = 1
TEST AH, E_CMD_FLAG ; Check Status Register bits 4 and 5 (high byte)
JNZ E_WRD_CMD_ERR ; Jump if = 1

TEST AL, E_ERR_FLAG ; Check Status Register bit 5 (low byte)
JNZ E_WRD_ERR ; Jump if = 1
TEST AH, E_ERR_FLAG ; Check Status Register bit 5 (high byte)
JNZ E_WRD_ERR ; Jump if = 1

TEST AL, VPP_FLAG ; Check Status Register bit 3 (low byte)
JNZ E_WRD_VPP ; Jump if = 1
TEST AH, VPP_FLAG ; Check Status Register bit 3 (high byte)
JNZ E_WRD_VPP ; Jump if = 1

E_WRD_ID_ERR:
; Insert code to service improper device ID read error here.
; Are 28F008SA RP# inputs disabled? Is Vcc applied to the 28F008SAs?
E_WRD_CMD_ERR:
; Insert code to service block erase command sequence error here
; (setup followed by a command other than confirm)
E_WRD_ERR:
; Insert code to service block erase error here
E_WRD_VPP:
; Insert code to service block erase Vpp low error here
E_WRD_CONT:
; Code continues from this point.....

```


This routine erases a block pair of two 28F008SAs. Note that all constants have been "OR'd" for parallel read/write of two devices at once. If a string of block pairs is to be erased at one time, the Vpp ramp up, RP# disable and device ID checks need only be done before the first block pair erase attempt. Additionally, when erasing multiple block pairs at once, examination of bits other than bit 7 (WSM Status) need only occur after the last block pair erase has completed. The Status Register retains any error bits until the Clear Status Register command is written.

3.0 'C' DRIVERS

```

/*****
/*      Copyright Intel Corporation, 1992
/*      Brian Dipert, Intel Corporation, May 7, 1992, Revision 2.1
/*      The following drivers control the Command and Status Registers of the 28F008SA Flash
/*      Memory to drive byte write, block erase, Status Register read and clear and
/*      array read algorithms. Sample Vpp and RP# control blocks are also included,
/*      as are example programs combining drivers into full algorithms
/*      The functions listed below are included:
/*      erasbgn(): Begins block erasure
/*      erassusp(): Suspends block erase to allow reading data from a block of the
/*      28F008SA other than that being erased
/*      erasres(): Resumes block erase if suspended
/*      end(): Polls the Write State Machine to determine if block erase or byte write
/*      have completed
/*      eraschk(): Executes full status check after block erase completion
/*      writebgn(): Begins byte write
/*      writechk(): Executes full status check after byte write completion
/*      idread(): Reads and returns the manufacturer and device IDs of the target
/*      28F008SA
/*      statrd(): Reads and returns the contents of the Status Register
/*      statclr(): Clears the Status Register
/*      rdmode(): Puts the 28F008SA in Read Array mode
/*      rdbyte(): Reads and returns a specified byte from the target 28F008SA
/*      vppup(): Enables high voltage Vpph
/*      vppdown(): Disables Vpph
/*      pwden(): Enables active low signal RP#
/*      pwddis(): Disables active low signal RP#
/*
/*      Addresses are transferred to functions as pointers to far bytes (ie long integers). An
/*      alternate approach is to create a global array the size of the 28F008SA and
/*      located "over" the 28F008SA in the system memory map. Accessing specific
/*      locations of the 28F008SA is then accomplished by passing the chosen function
/*      an offset from the array base versus a specific address. Different
/*      microprocessor architectures will require different array definitions; ie for
/*      the Intel architecture, define it as "byte eightmeg[16][10000]" and pass each
/*      function TWO offsets to access a specific location. MCS-96 architectures are
/*      limited to "byte eightmeg[10000]"; alternate approaches such as using port pins
/*      for paging will be required to access the full flash array
/*
/*      To create a far pointer, a function such as MK_FP() can be used, given a segment and
/*      offset in the Intel architecture. I use Turbo-C; see your compiler reference
/*      manual for additional information.
*****/

/*****
/*      Revision History: Rev 2.1
/*
/*      Changes From Revision 1.0 to Revision 2.0:
/*      Added alternate 28F008SA device ID to routine idread()
/*
/*      Changes from 2.0 to 2.1: Revised the Erase Suspend algorithm to remove potential
/*      "infinite loop" caused by the WSM going "ready" after the system reads the
/*      Status Register, and before the system issues the Erase Suspend command
*****/

typedef unsigned char byte;

```

```

/*****
/*      Function: Main
/*      Description: The following code shows examples of byte write and block
/*                  erase algorithms that can be modified to fit the specific application and
/*                  hardware design
*****/
main()
{
    byte far *address;
    byte data,status;

    /*
    /*      The following code gives an example of a possible byte write algorithm.
    /*      Note that Vpp does not need to be cycled between byte writes when a string of byte
    /*      writes occurs. Ramp Vpp to 12V before the first byte write and leave at 12V until after
    /*      completion of the last byte write. Doing so minimizes Vpp ramp up-down delay and
    /*      maximizes byte write throughput
    vppup();
    /*
    /*      "INSERT SOFTWARE DELAY FOR VPP RAMP IF REQUIRED"
    pwddis();
    address = 0XXXXXXL;
    data = 0Xyy;
    if (writebgn(data,address) == 1)
    /*
    /*      "RECOVERY CODE-POWER NOT APPLIED (ID CHECK FAIL)"
    else
    {
        while (end(&status) )
        {
            switch (writechk(status))
            {
                case 0:
                    break;
                case 1:
                    /*
                    /*      "RECOVERY CODE-VPP LOW DETECT ERROR"
                    break;
                case 2:
                    /*
                    /*      "RECOVERY CODE-BYTE WRITE ERROR"
                    break;
            }
            statclr();
        }
        vppdown();
    }
}

```

This "C" routine gives an example of combining lower-level functions (found in following pages) to complete a byte write. Routines vppup() and pwddis() enable the 28F008SA for byte write. Function writebgn() issues a byte write sequence to the device, end() detects byte write completion via Status Register bit 7, and writechk() analyzes Status Register bits 3-6 to determine byte write success. If a string of bytes is to be written at one time, Vpp ramp up and RP# disable need only be done before the first byte write attempt. Additionally, when writing multiple bytes at once, examination of bits other than bit 7 (WSM Ready) need only occur after the last byte write has completed. The Status Register retains any error bits until the Clear Status Register command is written.

```

/* The following code gives an example of a possible block erase algorithm. */
/* Note that Vpp does not need to be cycled between block erases when a string of block */
/* erases occurs. Ramp Vpp to 12V before the first block erase and leave at 12V until after */
/* completion of the last block erase. Doing so minimizes Vpp ramp up-down delay and */
/* maximizes block erase throughput */
vppup();
/* "INSERT SOFTWARE DELAY FOR VPP RAMP IF REQUIRED" */
pwddis();
address = 0XXXXXL;
if (erasbgn(address) == 1)
/* "RECOVERY CODE-POWER NOT APPLIED (ID CHECK FAIL)"
else
{
    while (end(&status) )
    {
        switch (eraschk(status))
        {
            case 0: break;
            case 1: "RECOVERY CODE-VPP LOW DETECT ERROR"
                    break;
            case 2: "RECOVERY CODE-BLOCK ERASE ERROR"
                    break;
            case 3: "RECOVERY CODE-ERASE SEQUENCE ERROR"
                    break;
        }
        statclr();
    }
    vppdown();
}

```

This "C" routine gives an example of combining lower-level functions (found in following pages) to complete a block erase. Routines vppup() and pwddis() enable the 28F008SA for block erase. Function erasbgn() issues a block erase sequence to the device, end() detects block erase completion via Status Register bit 7, and eraschk() analyzes Status Register bits 3-6 to determine block erase success. If a string of blocks is to be erased at one time, Vpp ramp up and RP# disable need only be done before the first block erase attempt. Additionally, when erasing multiple blocks at once, examination of bits other than bit 7 (WSM Ready) need only occur after the last block erase has completed. The Status Register retains any error bits until the Clear Status Register command is written.

```

/*
 * Description: Begins erase of a block.
 * Inputs:   blkaddr: System address within the block to be erased
 * Outputs:  None
 * Returns:  0 = Block erase successfully initiated
 *          1 = Block erase not initiated (ID check error)
 * Device Read Mode on Return: Status Register (ID if returns 1)
 */
*****

#define ERASETUP 0X20      /* Erase Setup command */
#define ERASCONF 0XD0     /* Erase Confirm command */

int erasbgn(blkaddr)
byte far *blkaddr;        /* blkaddr is an address within the block to be erased */
{
    byte mfgrid, deviceid;

    if (idread(&mfgrid, &deviceid) == 1) /* ID read error; device not powered up? */
        return (1);
    *blkaddr = ERASETUP; /* Write Erase Setup command to block address */
    *blkaddr = ERASCONF; /* Write Erase Confirm command to block address */
    return (0);
}

```

Routine erasbgn() issues a block erase command sequence to a 28F008SA. It is passed the desired system address for the block to be erased. After calling idread(), it writes the erase command sequence at the specified address. It returns "0" if block erase initiation was successful, and "1" if the ID read fails (device not powered up or RP# not disabled).


```

/*****
/*      Function: Erassusp
/*      Description: Suspends block erase to read from another block
/*      Inputs:   None
/*      Outputs:  None
/*      Returns:  0 = Block erase suspended
/*               1 = Error; Write State Machine not busy (block erase suspend not possible)
/*      Device Read Mode on Return: Status Register
*****/

#define RDYMASK  OX80 /* Mask to isolate the WSM Status bit of the Status Register */
#define WSMRDY   OX80 /* Status Register value after masking, signifying that
/*               the WSM is no longer busy
#define SUSPMASK OX40 /* Mask to isolate the erase suspend status bit of the
/*               Status Register
#define ESUSPYES OX40 /* Status Register value after masking, signifying that
/*               block erase has been suspended
#define STATREAD OX70 /* Read Status Register command
#define SYSADDR  0 /* This constant can be initialized to any address within
/*               the memory map of the target 28F008SA and is
/*               alterable depending on the system architecture
#define SUSPCMD  OXB0 /* Erase Suspend command

int erassusp()
{
    byte far *stataddr; /* Pointer variable used to write commands to device */

    stataddr = (byte far *)SYSADDR;
    *stataddr = SUSPCMD; /* Write Erase Suspend command to the device */
    *stataddr = STATREAD; /* Write Read Status Register command..necessary in case
/*               erase is already completed
    while ((*stataddr & RDYMASK) != WSMRDY)
    {
        /* Will remain in while loop until bit 7 of the Status
        /* Register goes to 1, signifying that
        /* the WSM is no longer busy
    }
    if ((*stataddr & SUSPMASK) = ESUSPYES)
    {
        return (0); /* Erase is suspended..return code "0"
    }
    return (1); /* Erase has already completed; suspend not possible.
/*               Error code "1"
}

```

Routine `erassusp()` issues the erase suspend command to a 28F008SA. It first makes sure the WSM is truly busy, then issues the erase suspend command and polls Status Register bits 7 and 6 until they indicate erase suspension. It returns "0" if block erase was successful, and "1" if the WSM was not busy when suspend was attempted.

```

/*****
/*      Function: Erasesres
/*      Description: Resumes block erase previously suspended
/*      Inputs:   None
/*      Outputs:  None
/*      Returns:  0 = Block erase resumed
/*               1 = Error; Block erase not suspended when function called
/*      Device Read Mode on Return: Status Register
*****/

#define RDYMASK  OX80 /* Mask to isolate the WSM Status bit of the Status Register */
#define WSMRDY   OX80 /* Status Register value after masking, signifying that the */
/*               WSM is no longer busy */
#define SUSPMASK OX40 /* Mask to isolate the erase suspend status bit of the */
/*               Status Register */
#define ESUSPYES OX40 /* Status Register value after masking, signifying that */
/*               block erase has been suspended */
#define STATREAD OX70 /* Read Status Register command */
#define SYSADDR  0 /* This constant can be initialized to any address within */
/*               the memory map of the target 28F008SA and is */
/*               alterable depending on the system architecture */
#define RESUMCMD  OXD0 /* Erase Resume command */

int erasres()
{
    byte far *stataddr; /* Pointer variable used to write commands to device */

    stataddr = (byte far *)SYSADDR;
    *stataddr = STATREAD; /* Write Read Status Register command to 28F008SA */
    if ((*stataddr & SUSPMASK) != ESUSPYES)
        return (1); /* Block erase not suspended. Error code "1" */
    *stataddr = RESUMCMD; /* Write Erase Resume command to the device */
    while ((*stataddr & SUSPMASK) == ESUSPYES)
        ; /* Will remain in while loop until bit 6 of the Status */
        /* Register goes to 0, signifying block */
        /* erase resumption */
    while ((*stataddr & RDYMASK) == WSMRDY)
        ; /* Will remain in while loop until bit 7 of the Status */
        /* Register goes to 0, signifying that the WSM is */
        /* once again busy */

    return (0);
}

```

Routine `erasres()` issues the erase resume command to a 28F008SA. It first makes sure the WSM is truly suspended, then issues the erase resume command and polls Status Register bits 7 and 6 until the indicate WSM resumption. It returns "0" if block erase resume was successful, and "1" if the WSM was not suspended when resumption was attempted.

```

/*****
/*      Function: End
/*      Description: Checks to see if the WSM is busy (is byte write/block erase completed?)
/*      Inputs:      None
/*      Outputs:     statdata: Status Register data read from device
/*      Returns:     0 = Byte Write/Block Erase completed
/*                  1 = Byte Write/Block Erase still in progress
/*      Device Read Mode on Return: Status Register
*****/
#define RDYMASK      0X80      /* Mask to isolate the WSM Status bit of the Status
#define WSMRDY       0X80      /* Register value after masking, signifying that the
                                /* WSM is no longer busy
#define STATREAD     0X70      /* Read Status Register command
#define SYSADDR      0         /* This constant can be initialized to any address within*
                                /* the memory map of the target 28F008SA and is
                                /* alterable depending on the system architecture */

int end(statdata)
byte *statdata;                /* Allows Status Register data to be passed back to the
                                /* main program for further analysis
{
    byte far *stataddr;         /* Pointer variable used to write commands to device
    stataddr = (byte far *)SYSADDR;
    *stataddr = STATREAD;       /* Write Read Status Register command to 28F008SA
    if (((*statdata = *stataddr) & RDYMASK) != WSMRDY)
        return (1);            /* Byte write/block erasure still in progress...code "1"
    return (0);                 /* Byte write/block erase attempt completed...code "0"
}

```

3

Routine end() detects completion of byte write or block erase operations of a 28F008SA. It passes back the Status Register data it reads from the device. It also returns "0" if Status Register bit 7 indicates WSM "Ready", and "1" if indication is that the WSM is still "Busy".

```

/*****
/*      Function: Erasechk
/*      Description: Completes full Status Register check for block erase (proper command
/*                  sequence, Vpp low detect, block erase success). This routine assumes that block
/*                  erase completion has already been checked in function end(), and therefore does
/*                  not check the WSM Status bit of the Status Register
/*      Inputs:   statdata: Status Register data read in function end
/*      Outputs:  None
/*      Returns:  0 = Block erase completed successfully
/*                1 = Error; Vpp low detect
/*                2 = Error; Block erase error
/*                3 = Error; Improper command sequencing
/*      Device Read Mode on Return: Same as when entered
*****/

#define ESEQMASK  0X30      /* Mask to isolate the erase and byte write status bits of */
                           /* the Status Register */
#define ESEQFAIL  0X30      /* Status Register value after masking if block erase */
                           /* command sequence error has been detected */
#define EERRMSK   0X20      /* Mask to isolate the erase status bit of the */
                           /* Status Register */
#define ERASERR   0X20      /* Status Register value after masking if block erase error */
                           /* has been detected */
#define VLOWMASK  0X08      /* Mask to isolate the Vpp status bit of the Status Register */
#define VPPLow    0X08      /* Status Register value after masking if Vpp low */
                           /* has been detected */

int erasechk(statdata)
byte statdata;              /* Status Register data that has been already read from the */
                           /* 28F008SA in function end() */

{
    if ((statdata & VLOWMASK) == VPPLow)
        return (1);        /* Vpp low detect error, return code "1" */
    if ((statdata & EERRMSK) == ERASERR)
        return (2);        /* Block erase error detect, return code "2" */
    if ((statdata & ESEQMASK) == ESEQFAIL)
        return (3);        /* Block erase command sequence error, return code "3" */
    return (0);            /* Block erase success, return code "0" */
}

```

Routine `erasechk()` takes the Status Register data read in `end()` and further analyzes it. It returns "0" if block erase was successful, "1" if Vpp low error was detected, "2" if block erase error was reported and "3" if an erase command sequence error was found (erase setup followed by a command other than erase confirm). This is useful after a block or string of blocks has been erased, to check for successful completion.


```

/*****
/*      Function: Writebgn
/*      Description: Begins byte write sequence
/*      Inputs:   wdata: Data to be written into the device
/*               waddr: Target address to be written
/*      Outputs:  None
/*      Returns:  0 = Byte write successfully initiated
/*               1 = Byte write not initiated (ID check error)
/*      Device Read Mode on Return: Status Register (ID if returns 1)
*****/

#define  SETUPCMD  0x40 /* Byte Write Setup command

int writebgn(wdata,waddr)

byte wdata;
byte far *waddr;

{
    byte mfgid,deviceid;

    if (idread(&mfgid,&deviceid)==1) /* Device ID read error...powered up?
        return (1);
    *waddr = SETUPCMD; /* Write Byte Write Setup command and destination address
    *waddr = wdata; /* Write byte write data and destination address
    return (0);

/*****
/*      Function: Writechk
/*      Description: Completes full Status Register check for byte write (Vpp low detect, byte
/*                  write success). This routine assumes that byte write completion has already
/*                  been checked in function end() and therefore does not check the WSM Status
/*                  bit of the Status Register
/*      Inputs:   statdata: Status Register data read in function end()
/*      Outputs:  None
/*      Returns:  0 = Byte write completed successfully
/*               1 = Error; Vpp low detect
/*               2 = Error; Byte write error
/*      Device Read Mode on Return: Status Register
*****/

#define  WERRMSK  0x10 /* Mask to isolate the byte write error bit of the
/*                    Status Register
#define  WRITERR  0x10 /* Status Register value after masking if byte write error
/*                    has been detected
#define  VLOWMASK 0x08 /* Mask to isolate the Vpp status bit of the
/*                    Status Register
#define  VPFLOW  0x08 /* Status Register value after masking if Vpp low
/*                    has been detected

int writechk(statdata)

byte statdata;

{
    if ((statdata & VLOWMASK) == VPFLOW)
        return (1); /* Vpp low detect error, return code "1"
    if ((statdata & WERRMSK) == WRITERR)
        return (2); /* Byte write error detect, return code "2"
    return (0); /* Byte/string write success, return code "0"
}

```

Routine writebgn() issues a byte write command sequence to a 28F008SA. It is passed the desired system address for the byte to be written, as well as the data to be written there. After calling idread(), it writes the byte write command sequence at the specified address. It returns "0" if byte write initiation was successful, and "1" if the ID read fails (device not powered up or RP# not disabled).

Routine writechk() takes the Status Register data read in end() and further analyzes it. It returns "0" if byte write was successful, "1" if Vpp low error was detected, and "2" if byte write error was reported. This is useful after a byte or string of bytes has been written, to check for successful completion.

```

/*****
/*      Function: Idread
/*      Description: Reads the manufacturer and device IDs from the target 28F008SA
/*      Inputs:      None
/*      Outputs:     mfgrid: Returned manufacturer ID
/*                  deviceid: Returned device ID
/*      Returns:    0 = ID read correct
/*                  1 = Wrong or no ID
/*      Device Read Mode on Return: Intelligent Identifier
*****/

#define MFGRADDR 0          /* Address "0" for the target 28F008SA...alterable depending */
/*                          on the system architecture */
#define DEVICADD 1          /* Address "1" for the target 28F008SA...alterable depending */
/*                          on the system architecture */
#define IDRDCOMM 0X90       /* Intelligent Identifier command */
#define INTELID 0X89        /* Manufacturer ID for Intel devices */
#define DVCID 0X0A2        /* Device IDs for 28F008SA */
#define DVCID2 0X0A1

int idread(mfgrid,deviceid)

byte *mfgrid;              /* The manufacturer ID read by this function, to be */
/*                          transferred back to the calling program */
byte *deviceid;            /* The device ID read by this function, to be transferred */
/*                          back to the calling function */

{
    byte far *tempaddr;     /* Pointer address variable used to read IDs */
    tempaddr = (byte far *)MFGRADDR;
    *tempaddr = IDRDCOMM;   /* Write Intelligent Identifier command to an address within */
/*                          the 28F008SA memory map (in this case 00 hex) */
    *mfgrid = *tempaddr;    /* Read mfgr ID, tempaddr still points at address "0" */
    tempaddr = (byte far *)DEVICADD; /* Point to address "1" for the device specific ID */
    *deviceid = *tempaddr;  /* Read device ID */
    if ((*mfgrid != INTELID) || ((*deviceid != DVCID) && (*deviceid != DVCID2)))
        return (1);        /* ID read error; device powered up? */
    return (0);
}

```

Routine `idread()` issues the Intelligent Identifier command to a 28F008SA. It passes back the manufacturer and device IDs it reads. In addition, it returns "0" if the IDs read matched those expected for the 28F008SA or 28F008SA-L, and "1" if either the manufacturer or device IDs did not match.

```

/*****
/*      Function: Statrd
/*      Description: Returns contents of the target 28F008SA Status Register
/*      Inputs:      None
/*      Outputs:     statdata: Returned Status Register data
/*      Returns:     Nothing
/*      Device Read Mode on Return: Status Register
*****/
#define  STATREAD  0X70          /* Read Status Register command
#define  SYSADDR   0            /* This constant can be initialized to any address within
                                /* the memory map of the target 28F008SA and is
                                /* alterable depending on the system architecture

int statrd(statdata)
byte *statdata;
{
    byte far *stataddr;          /* Pointer variable used to write commands to device
    stataddr = (byte far *)SYSADDR;
    *stataddr = STATREAD;        /* Write Read Status Register command to 28F008SA
    *statdata = *stataddr;
    return;
}

/*****
/*      Function: Statclr
/*      Description: Clears the 28F008SA Status Register
/*      Inputs:      None
/*      Outputs:     None
/*      Returns:     Nothing
/*      Device Read Mode on Return: Array
*****/
#define  STATCLR   0X50          /* Clear Status Register command
#define  SYSADDR   0            /* This constant can be initialized to any address within
                                /* the memory map of the target 28F008SA and is
                                /* alterable depending on the system architecture

int statclr()
{
    byte far *stataddr;          /* Pointer variable used to write commands to device
    stataddr = (byte far *)SYSADDR;
    *stataddr = STATCLR;        /* Write Clear Status Register command to 28F008SA
    return;
}

```

3

Routine statrd() reads a 28F008SA Status Register. It issues the Read Status Register command and passes back the data it obtains.

Routine statclr() issues the Clear Status Register command to a 28F008SA. This routine is required after analyzing Status Register contents in routines like eraschk() and writchk(). The 28F008SA Status Register retains state of bits 3–6 until they are cleared by the Clear Status Register command.

```

/*****
/*      Function: Rdmode
/*      Description: Puts the target 28F008SA in Read Array Mode. This function might be used, for
/*                  example, to prepare the system for return to code execution out of the Flash
/*                  memory after byte write or block erase algorithms have been executed off-chip
/*      Inputs:      None
/*      Outputs:     None
/*      Returns:     Nothing
/*      Device Read Mode on Return: Array
*****/

#define RDARRAY    0xFF          /* Read Array command
#define SYSADDR    0             /* This constant can be initialized to any address within
/*                               the memory map of the target 28F008SA and is
/*                               alterable depending on the system architecture

int rdmode()
{
    byte far *tempaddr;          /* Pointer variable used to write commands to the device
    tempaddr = (byte far *)SYSADDR;
    *tempaddr = RDARRAY;          /* Write Read Array command to 28F008SA
    return;

/*****
/*      Function: Rdbyte
/*      Description: Reads a byte of data from a specified address and returns it to the
/*                  calling program
/*      Inputs:      raddr: Target address to be read from
/*      Outputs:     rdata: Data at the specified address
/*      Returns:     Nothing
/*      Device Read Mode on Return: Array
*****/

#define RDARRAY    0xFF          /* Read array command

int rdbyte(rdata,raddr)
byte *rdata;                    /* Returns data read from the device at specified address
byte far *raddr;                /* Raddr is the target address to be read from

{
    *raddr = RDARRAY;           /* Write read array command to an address within the
/*                               28F008SA (in this case the target address)
    *rdata = *raddr;             /* Read from the specified address and store
    return;
}

```

Routine `rdmode()` simply puts a 28F008SA in Read Array mode. This is useful after byte write and block erase operations, to return the 28F008SA to its "normal" mode of operation. After block erase or byte write, the 28F008SA will continue to output Status Register data until the Read Array command is issued to it, for example.

Routine `rdbyte()` not only puts the 28F008SA in Read Array mode, it also reads a byte of data. It is passed the desired system byte address, and passes back the data at that address.


```

/*      Function: Vppup
/*      Description: Ramps the Vpp supply to the target 28F008SA to enable byte write or block
/*      erase. This routine can be tailored to the individual system architecture. For
/*      purposes of this example, I assumed that a system Control Register existed at
/*      system address 20000 hex, with the following definitions:
/*
/*      Bit 7: Vpph Control: 1 = Enabled
/*                          0 = Disabled
/*      Bit 6: PWD Control: 1 = PowerDown Enabled
/*                          0 = PowerDown Disabled
/*      Bits 5-0: Undefined
/*
/*      Inputs:  None
/*      Outputs: None
/*      Returns: Nothing
/*      Device Read Mode on Return: As existed before entering the function. Part is now ready for
/*      program or erase
/*
*****
#define  VPPHIGH  OX80      /* Bit 7 = 1, Vpp elevated to Vpph
#define  SYSCADDR OX20000   /* Assumed system Control Register Address

int vppup()
{
    byte far *contaddr;      /* Pointer variable used to write data to the system
                             /* Control Register

    contaddr = (byte far *)SYSCADDR;
    *contaddr = *contaddr | VPPHIGH; /* Read current Control Register data, "OR" with
                             /* constant to ramp Vpp

    return;

}

*****
/*      Function: Vppdown
/*      Description: Ramps down the Vpp supply to the target 28F008SA to disable byte write/block
/*      erase. See above for a description of the assumed system Control Register.
/*
/*      Inputs:  None
/*      Outputs: None
/*      Returns: Nothing
/*      Device Read Mode on Return: As existed before entering the function. Part now has high Vpp
/*      disabled. If byte write or block erase was in progress when this function was
/*      called, it will complete unsuccessfully with Vpp low error in the
/*      Status Register.
/*
*****
#define  VPPDWN  OX7F      /* Bit 7 = 0, Vpp lowered to Vppl
#define  SYSCADDR OX20000   /* Assumed system Control Register Address

int vppdown()
{
    byte far *contaddr;      /* Pointer variable used to write data to the system
                             /* Control Register

    contaddr = (byte far *)SYSCADDR;
    *contaddr = *contaddr & VPPDWN; /* Read current Control Register data, "AND" with
                             /* constant to lower Vpp

    return;

}

```

Functions vppup() and vppdown() give examples of how to control via software the hardware that enables or disables 12V V_{pp} to a 28F008SA. The actual hardware implementation chosen will drive any modification of these routines.

```

/*****
/*      Function: Pwden
/*      Description: Toggles the 28F008SA RP# pin low to put the device in Deep PowerDown mode.
/*                  See above for a description of the assumed system Control Register.
/*      Inputs:      None
/*      Outputs:     None
/*      Returns:     Nothing
/*      Device Read Mode on Return: The part is powered down. If byte write or block erase was in
/*                  progress when this function was called, it will abort with resulting partially
/*                  written or erased data. Recovery in the form of repeat of byte write or block
/*                  erase will be required once the part transitions out of powerdown, to
/*                  initialize data to a known state.
*****/

#define PWD      OX40          /* Bit 6 = 1, RP# enabled
#define SYSCADDR OX20000      /* Assumed system Control Register Address

int pwden()
{
    byte far *contaddr; /* Pointer variable used to write data to the system
                        /* Control Register

    contaddr = (byte far *)SYSCADDR;
    *contaddr = *contaddr | PWD; /* Read current Control Register data, "OR" with constant
                                /* to enable Deep PowerDown

    return;
}

/*****
/*      Function: Pwddis
/*      Description: Toggles the 28F008SA RP# pin high to transition the part out of Deep
/*                  PowerDown. See above for a description of the assumed system Control Register.
/*      Inputs:      None
/*      Outputs:     None
/*      Returns:     Nothing
/*      Device Read Mode on Return: Read Array mode. Low voltage is removed from the RP# pin.
/*                  28F008SA output pins will output valid data time tPHQV after the RP# pin
/*                  transitions high (reference the datasheet AC Read Characteristics) assuming
/*                  valid states on all other control and power supply pins.
*****/

#define PWDOFF   OXBF          /* Bit 6 = 0, RP# disabled
#define SYSCADDR OX20000      /* Assumed system Control Register Address

int pwddis()
{
    byte far *contaddr; /* Pointer variable used to write data to the system
                        /* Control Register

    contaddr = (byte far *)SYSCADDR;
    *contaddr = *contaddr & PWDOFF; /* Read current Control Register data, "AND" with
                                /* constant to disable Deep PowerDown

    return;
}

```

Functions `pwden()` and `pwddis()` give examples of how to control via software the hardware that enables or disables a 28F008SA RP# input. The actual hardware implementation chosen will drive any modification of these routines.

ADDITIONAL INFORMATION

		Order Number
	28F008SA Datasheet	290429
	28F008SA-L Datasheet	290435
AP-359	"28F008SA Hardware Interfacing"	292094
AP-364	"28F008SA Automation and Algorithms"	292099
ER-27	"The Intel 28F008SA Flash Memory"	294011
ER-28	"ETOX-III Flash Memory Technology"	294012

REVISION HISTORY

Number	Description
002	Revised Erase Suspend Algorithm in "C" Drivers.
003	PWD # pin renamed RP # to match JEDEC standards.

ADDITIONAL INFORMATION

284028	"284028A Flash Memory Technology"
284011	"284028A Flash Memory"
283089	"284028A Automation and Algorithms"
283084	"284028A Hardware Interfacing"
280438	"284028A Datasheet"
280430	"284028A Datasheet"

REVISION HISTORY

Revision	Description
003	Revised Erase Suspend Algorithm in "C" Driver
002	PWD = pin renamed RP to match JEDEC standards

Implementing Mobile PC Designs Using High Density FlashFile™ Components

3

DON VERNER
SENIOR APPLICATIONS ENGINEER

October 1993

IMPLEMENTING MOBILE PC DESIGNS USING HIGH DENSITY FlashFile™ COMPONENTS

CONTENTS	PAGE
1.0 INTRODUCTION	3-141
1.1 Why A New Memory Architecture	3-141
1.2 The Flash Memory Alternative ...	3-142
2.0 SOFTWARE DESIGN	3-143
2.1 XIP Operating System	3-143
2.2 DOS In Flash Implementation	3-144
2.3 Resident Flash Disk	3-146
2.3.1 Microsoft's Flashfile System	3-147
2.4 Resident Flash Disk (RFD) and ExCA Architecture	3-147
2.4.1 ExCA Software Interface	3-148
2.4.2 RFD Socket Services	3-150
2.4.3 ExCA Card Services	3-150
2.4.4 iCARD29 File System Driver	3-150
2.4.5 RFD Socket Services	3-151
2.5 XIP Graphical Users Interface (GUI) Overview	3-151
2.6 XIP GUI Implementation	3-152
2.7 Pen Extensions	3-153
3.0 HARDWARE	3-153
3.1 Resident Flash Disk Implementation	3-153
3.2 XIP DOS Implementation	3-155

CONTENTS	PAGE
3.3 XIP GUI Implementation	3-157
3.4 Chipset Considerations	3-157
3.5 SL Based Design	3-157
3.6 Schematic Overview	3-158
4.0 SOFTWARE UTILITIES	3-159
4.1 RFA Diagnostic	3-159
4.2 RFA Binary Loader	3-160
5.0 SUMMARY	3-160
APPENDIX A: Additional Publications	3-161
APPENDIX B: MS DOS ROM Image Description	3-163
APPENDIX C: ROMDISK CONFIG.SYS and AUTOEXEC.BAT	3-165
APPENDIX D: Windows 3.1 CONTENTS.ROM	3-167
APPENDIX E: EPLD Equations	3-173
APPENDIX F: RFA Schematics	3-185

1.0 INTRODUCTION

As personal computers migrate to platforms that are easily held with one hand, a revolutionary system architecture is required to meet space and power requirements.

- An architecture that is not bounded by what has been done before with existing memory architecture, but free to meet the demanding requirements of mobile end users.
- An architecture free to adapt and accommodate new technological advances in software and hardware, while protecting end-users initial base hardware investment.

Implementing this new system architecture requires traditional PC storage media such as ROM, DRAM, floppy disk and hard disk to move aside and make room for the latest in memory storage, Intel's FlashFile™ memory (see architecture comparison in Figure 1.).

Application	Data Manipulation	Code Execution	File & Code Storage
Desktops	DRAM	DRAM/ROM	FDD/HDD
Portables	DRAM	FLASH	FLASH - Resident Disk - Flash Card - Flash Drive

292097-1

Figure 1. Architecture Comparison

By combining FlashFile memory with new system architecture, completely new types of computers are now possible that fit in the palm of your hand and replace or integrate many of the code or storage functions of other memory types. Moreover, FlashFile memory enables hand-held computers to last many hours on just a couple of AA batteries. FlashFile memory can be used for storing eXecute-In-Place (XIP) code in the system's memory map, while additionally functioning like a disk for file and program storage. Since this type of design features FlashFile memory resident on the PC's motherboard and is typically arranged in an array, it is described as a Resident Flash Array (or RFA). To further differentiate the two tasks of an RFA, the file store task is called a Resident Flash Disk (RFD), while the XIP task is called Resident Flash for XIP (or RFX) code storage.

The FlashFile memory is also used in card form as specified by the Personal Computer Memory Card International Association (PCMCIA). Flash memory cards provide file and program storage similar to an RFD, but add the feature of removability, increasing or adding to ease-of-use for the end user. The PCMCIA specification addresses both the memory and I/O card's physical, electrical and mechanical characteristics, while leaving the host PC implementation relatively free for interpretation. Enhancing the PCMCIA specification, Intel developed the Exchangeable Card Architecture (ExCA™), which defines the host PC system card interface. ExCA further refines the PCMCIA specification and provides for card exchangeability and inter-operability for both memory and I/O cards.

Memory and I/O cards complement this new mobile architecture by integrating many of the common but functionally separate tasks used by today's mobile professional in either electronic or paper form. Some of these tasks are schedule keepers, phones, address books, checkbooks, credit cards, fax, pagers, personal voice storage, task managers/schedulers, paperless form reports, scratch pads, and notepads.

1.1 Why A New Memory Architecture?

The ideal hand held memory system is:

- Power Conscious (prolongs battery life and reduces heat)
- Dense (stores lots of code and data in a small amount of space but weighs very little)
- Updateable (allows in-situ code enhancements)
- Fast (lets you read and write data quickly)
- Inexpensive (low cost per megabyte)
- Reliable (retains data when exposed to extreme temperature and mechanical shock)

Since the PC's introduction over 10 years ago, designers have grappled with how to construct memory systems that met the above criteria. Portable computing makes the system design even tougher with more stringent requirements for low power, low volume and less weight. The best combination available for portable designs in their infancy was the same as used for the desktop; solid-state memory and magnetic storage, i.e., SRAMs, DRAMs plus magnetic hard disks. DRAMs are dense and inexpensive, yet slower than the processors they serve, and they are volatile. SRAMs, although fast enough to keep pace with processors, are relegated to caching schemes (compensating for DRAM's slowness) due to low density and high cost while also being volatile. Magnetic hard disks, the nonvolatile append-

age to DRAM and SRAM, are dense, inexpensive on a cost-per-megabyte basis and nonvolatile, but are slow, power hungry, take up a sizable amount of volume and are susceptible to damage from physical shock.

Mobile computing designs cannot depend on hard drives as portable notebook PCs do. Volume (4" x 8 x 0.5" or less), power (two AAs), and shock constraints preclude using even the 1.3" hard drives available today. Also, vitally important data such as credit card numbers or transactions, signatures, or checkbook information demands reliability of the highest order.

1.2 The Flash Memory Alternative

High Density

Intel's ETOX III Flash Memory Cell is 30% smaller than equivalent DRAM cells; therefore it will closely track DRAM density. Intel's 28F008SA FlashFile Memory can store 8 megabits, or one megabyte, of data. Flash memory is more scaleable than DRAM because the flash storage cell is not sensitive to soft error failure, therefore it can have a more simple cell structure. Thus as density increases and process lithography continues to shrink, flash memory will pace and ultimately overtake the DRAM technology treadmill.

Updatable

ROMs and EPROMs may offer lower device costs, but if servicing the customer or end-user is important to an OEM, overall system cost must be factored in. Although ROMs and EPROMs are nonvolatile, changing the code within them is either very difficult (in the case of EPROMs) or entirely impossible (in the case of ROMs). Whole inventories of ROMs could be lost in the event of a catastrophic bug, while an innovative design with FlashFile memory can be updated in the factory or by end-users via networks, OEM Bulletin Board Systems, or other memory cards. Updating systems could actually become a second source of income for OEMs and Independent Software Vendors, enhancing the quality of the product while increasing end-user satisfaction.

Power Conscious

Intel's FlashFile memory provides a deep powerdown mode, reducing power consumption to typically less than 0.2 μ A. Typical read current is only 20 mA while typical standby current (flash memory not being accessed with CE# high) is only 30 μ A. Additionally, FlashFile devices operating at 3.3 V_{CC} are available for state-of-the-art low power consumption designs.

Fast

Don't be misled by technology-to-technology speed comparisons. Architecting a system around FlashFile memory bypasses the code/data bottleneck created by connecting slow mechanical serial memory (such as disks) to a high-performance, parallel-bussed processor system. For example, data seek time for a 1.8" magnetic hard disk is 20 ms, plus an 8 ms average rotational delay, while flash memory access time is less than 0.1 ms. At the chip level, read speeds for FlashFile memory are about 85 ns. Therefore, either direct execution of code from flash memory or downloading to system RAM will dramatically enhance overall system performance.

Nonvolatile

Unlike DRAM or SRAM, FlashFile memory requires no battery backup. Further, Intel's flash devices retain data typically for over 100 years, well beyond the useful lifetime of even the most advanced computer.

Rugged and Reliable

On average, today's hard-disk drives can withstand up to 10 Gs of operating shock; Intel's FlashFile memory can withstand as much as 1000 Gs. FlashFile components can operate beyond 70°C while magnetic drives are limited to 55°C. Intel's FlashFile Memory can be cycled 100,000 times per block or segment. Even beyond that cycle level, FlashFile does not fail or lock up like EEPROM devices, it just tends to take longer to erase blocks and program bytes than the times specified in the data sheet. By employing wear-leveling techniques, a 10 KB file written every 5 minutes, 24 hours a day to a 20 MB flash array takes 1.2 million hours or 136 years before reaching the 10,000 cycle level.

Many applications benefit from ROMed or XIP versions of code, particularly hand held personal computers, vertical application pen-based clipboards, and industrial control and data accumulation equipment. These applications pose system design constraints requiring small form factor, low power consumption, and ruggedized construction due to active mobile users or harsh environments. Exposure to shock, vibration, or temperature extremes is common, precluding the use of rotating media. Flash memory provides an excellent code storage choice for such system designs featuring thin TSOP packaging, low (deep powerdown mode) or zero (capability to shut off power without losing data) power consumption, and 1000 G shock resistance and extended temperature products. Additionally flash memory provides remote or end-user update capability over ROMs, allowing OEM's to service their products more efficiently and add new software features and applications after the sale.

new, compact and portable system architecture. This application note discusses implementing just such a design using Intel's boot block flash memory and Intel's FlashFile (28F008SA) components for extended memory eXecute-In-Place (XIP) code store, disk-like functionality for file and program storage, and BIOS code storage.

Related Publications

The following data books and reference manuals provide valuable information for developing an RFA-based design:

- Intel 28F008SA Data Sheet Order number 290429
- Microsoft MS-DOS 5.00 ROM Technical Specification and OEM Adaptation Kit (OAK)
- Microsoft Flash Filing System OAK
- Microsoft Windows 3.1 ROM Technical Specification and OAK

For additional information on flash memory, power supply solutions and EPLDs, see Appendix A.

2.0 SOFTWARE DESIGN

Software design is considered first for solid-state designs because the software functionality desired affects in large part how the hardware design is implemented. Many software products exist for solid-state systems:

- DOS operating systems from Microsoft, Novell's Palm DOS and Datalight's ROM-DOS
- A Graphical User Interface (GUI) operating system in Windows 3.1 ROM version
- Application software from Microsoft (such as Word, Excel and MS-Works) and Lotus 1-2-3 Version 2.2.

Because no standardization exists, implementation differs from package to package or vendor to vendor. Therefore, this application note describes a system using MS DOS ROM Version, MS Flash Filing System, Windows 3.1 ROM Version plus Pen extensions. All are readily available applications today and offer the highest inter-compatibility. However, the hardware and software design concepts presented here work just as well with Novell's (formally DRI) Palm DOS and also GO Corporation's Penpoint operating systems.

2.1 XIP Operating System

The first decision one must make for a solid-state software design is the operating system. Many alternatives exist for small hand held computer systems. Any solution depends on what requirements are placed on desktop compatibility, software compatibility, and ease of

compatibility is not necessarily a requirement. This is evident by the multiple emerging entries of pen-based GUI's. However, data transfer using a medium such as memory cards between desktop systems and hand held computers depends on an agreed file format. At least for now, DOS is still the major operating system of choice for the largest number of desktop systems. Therefore, DOS compatibility is still a necessity for many hand held computing systems and is incorporated in this design.

To insure compatibility and easy system integration, MS DOS 5.0 ROM Version is the easiest choice. With this version, an XIP DOS implementation can be configured to as small as 64 KB using just the Runtime Kernel and a minimized command interpreter. If CONFIG.SYS and AUTOEXEC.BAT processing is required, an additional 56 KB are required plus a ROM DISK large enough to hold AUTOEXEC.BAT, CONFIG.SYS, and any drivers and files that are referenced by either file.

Microsoft provided the capability for additional XIP DOS applications to be added to an MS DOS XIP implementation by providing two new DOS functions, "Find First ROM Program" and "Find Next ROM Program." This allows DOS-based XIP applications (such as OEM-specific utilities and applications or Embedded DOS applications) to easily be added to the MS DOS 5 ROM build.

Many different memory configurations MS-DOS ROM are possible by distributing various software pieces (Microsoft refers to them as granules) between different ROM locations below and above the 1 MB address space. Certain restrictions exist on individual granules requiring them to appear below 1 MB. The granules and address location requirements are specified in a table within the MS DOS ROM 5.0 OAK. Approximately 43 KB of granules must be located below the first 1 MB of address space. Other granules can be located either below or above 1 MB. The total size of all granules in this design is approximately 128 KB.

How MS DOS ROM Boots Up

For system startup and booting DOS without a disk, MS DOS 5 ROM must intercept the INT 19h call made by the BIOS. This is accomplished by locating a granule as an adapter ROM within adapter space (C0000h-EFFFFh). This granule contains the ROM scan identifier "55AAh" which must appear on a 2 KB boundary and identifies the module as a ROM to the BIOS during Power On Self-Test (POST), and also identifies the MS DOS 5 ROM INT 19h interceptor. When the BIOS Post code identifies the ROM, control is turned over to the ROM for its initialization. At this time, the MS DOS 5 ROM redirects the INT 19h vector to the MS DOS 5 ROM code and control returns back to the

BIOS POST code. When the BIOS is completely finished testing and initializing, it issues INT 19h, and the MS DOS 5 ROM INT 19h handler gains control. The handler loads the MS DOS 5 ROM bootstrap loader into RAM and passes control to it. If the bootstrap loader includes the "Multi-Boot" option, a list of menu boot options are presented to the user if an OEM defined key is being pressed. The menu might look like Figure 2.

Booting from a disk:	
1.	Boot from Floppy Disk.
2.	Boot from Hard Disk
Booting from ROM:	
3.	Floppy is default; process startup files.
4.	Hard drive is default; process startup files.
5.	Floppy is default; do not process startup files.
6.	Hard drive is default; do not process startup files.
7.	ROM drive is default; process startup files.

Figure 2. Multi-Boot Menu

The menu is activated by pressing the ALT key during the system memory scan, which is the default provided in the OAK. Other types of keys may be selected by the OEM for their specific implementation. Selecting the options under Booting from a disk will bypass the ROM system completely. Selecting the options under Booting from ROM will invoke MS DOS 5 ROM and whatever the option specifies for processing the startup files, CONFIG.SYS and AUTOEXEC.BAT.

If the Multi-Boot option is not available or is not activated by the user, the MS DOS 5 ROM bootstrap loader reads a byte from CMOS RAM to determine boot options. The byte is defined in Figure 3.

Reserved	Default Drive If ROM Boot	Default Drive If ROM Boot	ROM CONFIG.SYS Processing	Default Boot
7 to 4	3	2	1	0
Default Boot Bit 0:				
0- Boot from ROM				
1- Normal disk boot operation (first is drive 00, then drive 80)				
ROM Configuration Processing Bit 1:				
0- Process CONFIG.SYS and AUTOEXEC.BAT from default drive				
1- Do NOT process CONFIG.SYS or AUTOEXEC.BAT files				
Default Drive if ROM Boot Bits 2 & 3:				
00- First Floppy Drive (0h)				
01- First Hard Drive (80h)				
10- ROM Drive				

Figure 3. CMOS Byte Definitions

If the system is to boot from ROM by either selecting the Multi-Boot option or reading the CMOS byte, the MS DOS 5 ROM bootstrap loader loads BIOS (note: not system BIOS, but the BIOS layer of MS DOS) and DOS initialization granules into RAM, records the addresses of the resident BIOS and DOS code granules in the BIOS data area, records boot options (default drive, CONFIG.SYS & AUTOEXEC.BAT processing) in the BIOS data area, and passes control to BIOS initialization. Just before the end of BIOS initialization, control is passed to SYSINIT which moves itself and DOS data and initialization code to high memory where both SYSINIT and DOS initialization takes place. Next SYSINIT then reads and processes the CONFIG.SYS file where installed drivers called in CONFIG.SYS and additional elements of the DPB chain will be placed in memory following the existing DOS structures as well as system buffers allocated. At this point, system bootstrap is finished and the command interpreter is started using the DOS call to execute ROM utilities. If a full command processor is chosen, the user will now see "C:>" prompt.

2.2 DOS In Flash Implementation

For this particular implementation, a full version of MS DOS with the full Command processor was chosen. This configuration uses 64 KB of adapter space (upper memory) at E0000h to EFFFFh for MS-DOS 5 ROM, and a combination 64 KB XIP binary file and a 256 KB ROM Disk binary file located in extended memory at F90000h. See Figure 4. ROM DOS and ROM Disk Memory Maps. This location is just after the end of the XIP GUI code block. The 64 KB XIP Binary contains the transient portion of COMMAND.COM and the DOS BIOS initialization and is created during the build of the MS DOS 5 ROM system, while the 256 KB ROM Drive contains the necessary files for bringing up the system and loading the MS Flash File System drivers which is addressed in the next section.

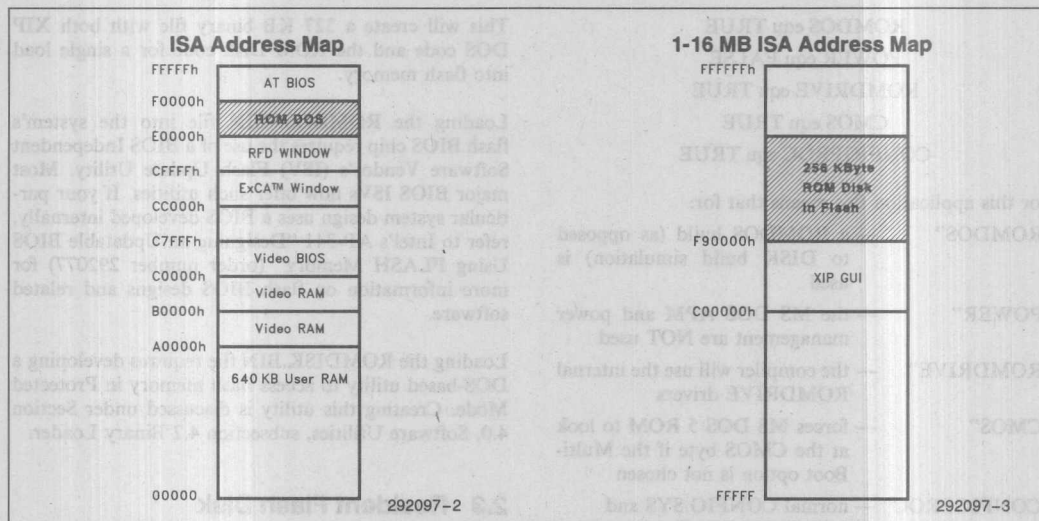


Figure 4. ROM DOS and ROM Disk Memory Maps

The E0000h segment was chosen because it happens to be free on the flash BIOS storage chip. Also, later on in this design we will be adding an RFD, which uses up the whole D0000h segment, and ExCA which requires 16 KB of adapter space that we located at CC000h (see Figure 4 for a memory map). The ROM disk image location was chosen because there was room available in the extended XIP portion. The XIP portion is located at C00000h (12 MB) due to compatibility reasons with the 386SL processor and SL window constraints. In a non-SL design, the ROM disk portion of MS DOS ROM can be located anywhere in extended memory above 1 MB.

A copy of the ROM image description file is included in Appendix B. The following summarizes the steps

taken for building a ROM version of DOS is taken from the MS-DOS 5.00 ROM OAK. Please refer to the MS-DOS 5.00 ROM OAK for specific details.

There are a set of compile options specified in the MS DOS 5 ROM OAK that need to be defined by the OEM for their particular implementation and are contained in the OAK file named "VERSION.INC". The compile options used for this example are listed as follows:

3

```

ROMDOS equ TRUE
POWER equ FALSE
ROMDRIVE equ TRUE
CMOS equ TRUE
CONFIGPROC equ TRUE

```

For this application this means that for:

- "ROMDOS" — a ROMDOS build (as opposed to DISK build simulation) is used
- "POWER" — the MS DOS APM and power management are NOT used
- "ROMDRIVE" — the compiler will use the internal ROMDRIVE drivers
- "CMOS" — forces MS DOS 5 ROM to look at the CMOS byte if the Multi-Boot option is not chosen
- "CONFIGPROC" — normal CONFIG.SYS and AUTOEXEC.BAT processing will be used

Since we are planning to use a ROM DRIVE, MS DOS 5 ROM needs to know where the ROM DRIVE exists. To set the ROM drive base address, the MS DOS 5 ROM OAK file ROMRDHI.ASM must be modified. Edit the file and set the ROMDRIVERBASE_LO equal to 0000h and the ROMDRIVERBASE_HI equal to 0FAh, or 64 KB above the base address of the extended memory XIP module.

Now the MS DOS ROM binaries are ready for building. Using the MS NMK utility, 3 separate binary files are compiled based on the requirements and addresses specified in the MS-DOS ROM Image Description file in Appendix B. ROM binary files 1 and 2 can be combined into one 64 KB binary image as follows:

```
copy /b ROM1.BIN + ROM2.BIN ROMDOS5.BIN
```

Once the XIP binaries are built, the rest of the ROM disk needs to be built. First, specify a RAM Drive within your build or development PC using the MS DOS RAMDRIVE.SYS device driver as follows:

```
device = RAMDRIVE.SYS 256 /e
```

After rebooting your PC, copy the files needed for your particular application. For an example of CONFIG.SYS and AUTOEXEC.BAT files, see Appendix C. Change the drive label as per the MS DOS 5 OAK instructions and then use the MS DOS IMGET utility to capture the image of the RAM drive into a binary file. Next, concatenate the ROM3.BIN binary image created by the NMK with the ROM Disk image captured from the RAM drive by using the MS DOS copy command as stated earlier:

```
copy /b ROM3.BIN + RDISK.BIN ROMDSK.BIN
```

This will create a 327 KB binary file with both XIP DOS code and the ROM Disk code for a single load into flash memory.

Loading the ROMDOS5.BIN file into the system's flash BIOS chip requires the use of a BIOS Independent Software Vendor's (ISV) Flash Update Utility. Most major BIOS ISVs now offer such utilities. If your particular system design uses a BIOS developed internally, refer to Intel's AP-341 "Designing an Updatable BIOS Using FLASH Memory" (order number 292077) for more information on flash BIOS designs and related software.

Loading the ROMDISK.BIN file requires developing a DOS-based utility to access flash memory in Protected Mode. Creating this utility is discussed under Section 4.0, Software Utilities, subsection 4.2 Binary Loader.

2.3 Resident Flash Disk

Once a DOS-based, XIP operating system is in place, the next area to work on is file storage for flash memory. File storage is possible with either FlashFile components or FlashFile cards, since they appear the same to file system software. However, the characteristics of flash memory are very unlike magnetic storage media characteristics. File Allocation Table (FAT)-based systems rely on the fact that the operating system has unrestricted write capability and/or access to the media, particularly when updating the FAT for a file creation, update or deletion. Flash memory on the other hand, does not necessarily allow write access 100% of the time. When the flash memory media is completely erased (all FFh's), writing data to the media can occur at any time and at any location. Additional data writes within the same block but at different locations can also occur. However, once a bit is written to a zero (0b), erasure of the whole block is required before allowing that particular bit to change back to one (1b). This asymmetrical characteristic of flash memory prevents using a straight implementation of the FAT-based file architecture and requires an alternative file system implementation to take advantage of Flash Memory benefits.

Blocking

Intel's FlashFile memory offers 64 KB separately erasable blocks enhancing the use of flash memory as a file storage medium. The large block size (as opposed to 512 byte blocks or sectors) provides the system with fewer total blocks to manage, resulting in less system overhead used for file management. Additionally, large blocks reduce file fragmentation since large files will most likely be contained in one block as opposed to several 512 byte sectors. This reduced fragmentation

file system. These features led to the creation of Microsoft's Flash Filing System.

2.3.1 MICROSOFT'S FLASH FILING SYSTEM

To enhance the use of flash memory as a disk, Microsoft created the Flash Filing System. This file system operates as a list of linked lists while keeping track of individual block erasure and file deletions, using minimal system overhead. File allocation structures use indirectly linked lists, allowing the file system to update files and data within the files without first requiring the area where the file is located to be erased and then updated. During file deletion, a file's header structure is written to mark the file as deleted, removing the file from the file allocation listing. Once the array of flash memory contains a majority of deleted files, the file system performs a (background) cleanup operation and copies good files out to free blocks and erases the blocks with all the deleted files. This achieves the goal of the user being able to use flash memory the same as they would use any other mass storage media without doing anything different.

Three distinct parts comprise the file system organization and implementation:

- A File System Redirector, whose job is to intercept the disk operations passed to MS DOS by an application and translate them into generic file operations, passing them on to the file system driver.
- A File System Driver, which accepts generic file operations passed to it from the File System Redirector, implements the architecture and logic of the Microsoft Flash Filing System, and passes low-level commands such as read, write, copy, and erase to the device driver.
- A Device Driver, which accepts low-level commands from the File System Driver and interfaces to the host system hardware implementation.

The File System Redirector performs a task analogous to a network redirector for LAN (Local Area Network) systems and appends itself to MS DOS. Any applications then think they are running from a networked file. Some classes of applications and utilities will not operate via this interface. Specifically, those applications that issue the INT 13h disk BIOS I/O call, INT 25h DosAbsRead, or INT 26h DosAbsWrite calls will not work properly with the Flash Filing system, just as they would not work over a network LAN. The File System Driver treats the flash media as a collection of large blocks, all identical in size. Individual block statistics are kept within a variable length structure at the top of

directory, the control structures and the data storage. The File System Driver also determines when de-allocated space (deleted files or directories) within a block is reclaimed for re-use.

The device driver portion is OEM-modifiable and needs to be written for the specific hardware example used. This device driver, specifically called iCARD29, is covered in more detail in the next section, ExCA Architecture, under iCARDDRV File System Driver Summary. The only MS FlashFile System hardware requirement is a single window available per socket in a system's adapter space that addresses all the flash memory to be used. Window size and base address are left to the system designer to decide, based on system design requirements. Some hardware guidelines are:

- Register-defined window size of either 4 KB, 8 KB, 16 KB, or 32 KB
- Register-defined base address in adapter space (C0000h to DFFFFh)

For more detailed information on Microsoft's Flash Filing System, consult the Microsoft Flash Filing System OAK.

3

2.4 Resident Flash Disk (RFD) and ExCA Architecture

Many systems which use an RFA will also want to incorporate PCMCIA memory and I/O cards. If an RFD uses the same software architecture used for PCMCIA cards, less software duplication is present in systems containing both cards and RFDs. This section discusses the Intel Exchangeable Card Architecture (ExCA) as it applies to an RFD and a Flash Filing system.

Most of this section was excerpted from the ExCA 1.1 Specification. The reader is encouraged to obtain that document for more details not revealed in this discussion. Other documents are the PCMCIA PC Card Standard Release 2.0, the PCMCIA Card Services Interface Specification, and the PCMCIA Socket Services Interface Specification Release 2.0.

ExCA specifies a standard host system hardware and software interface for 68 pin, PCMCIA/JEIDA memory and I/O cards. ExCA Release 1.10 defines the minimum hardware and software interfaces that card and system designers can rely on for basic compatibility across PC Cards, systems, and related software. By defining these interfaces, ExCA makes the PCMCIA goal of PC Card inter-operability a reality.

2.4.1 ExCA SOFTWARE INTERFACE

The primary purpose of the ExCA Software Interface is to explicitly define a minimal set of socket control and resource access functions upon which higher-level PC Card Client device drivers can rely. A PCMCIA implementor may incorporate a range of functions beyond the basic Memory Card Interface specified in PCMCIA 1.0. For PCMCIA 2.0, three primary functional extensions to the specification exist. They are: I/O devices, L-XIP-mapped memory ("L" stands for LIM or Lotus, Intel, Microsoft), and E-XIP-mapped memory ("E" stands for Extended or Protected mode memory). While basic memory requirements can be met with a single, small memory-mapped window or even via an I/O approach, both XIP modes require direct-mapping interface capability, with very specific boundaries in the L-XIP mode. Without an ExCA-like hardware and software support for direct-mapped memory, XIP-formatted cards cannot function. The ExCA socket hardware and software specifications define basic, clear compatibility definitions for PC cards, software drivers, and host systems.

ExCA allows PC Cards and sockets to be accessed by multiple PCMCIA-aware device drivers, configuration utilities and applications, with efficient and non-conflicting use of system resources. An architectural diagram of ExCA functionality is shown in Figure 5. The primary components of the software interface are Socket Services and Card Services. Socket Services provides the lowest-level function set for socket hardware adapter control. Card Services allocates resources and coordinates PC Card-related activities for higher-level client device drivers.

2.4 Resident Flash Disk (RFD) and ExCA Architecture

Many systems which use an RFD will also want to incorporate PCMCIA memory and I/O cards. If an RFD uses the same software architecture used for PCMCIA cards, less software duplication is present in systems containing both cards and RFDs. This section describes the Intel Exchangeable Card Architecture (ExCA) as it applies to an RFD and a Flash Filing System.

Most of this section was excerpted from the ExCA 1.1 Specification. The reader is encouraged to obtain that document for more details not revealed in this document. Other documents are the PCMCIA PC Card Standard Release 2.0, the PCMCIA Card Services Interface Specification, and the PCMCIA Socket Services Interface Specification Release 2.0.

ExCA specifies a standard host system software and software interface for 68 pin PCMCIA/EISA memory and I/O cards. ExCA Release 1.0 defines minimum hardware and software interfaces that card and system designers can rely on for basic compatibility across PC Cards, systems, and related software. By defining these interfaces, ExCA makes the PCMCIA goal of PC Card interoperability a reality.

Individual block hardware and the device driver software use the allocation structures use in the system to allow the file system to update files and data without the file system first regarding the file size where the file is located to be created and then updated. During the deletion, a file's header structure is written to mark the file as deleted, removing the file from the file allocation listing. Once the array of flash memory contains a majority of deleted files, the file system performs a (background) cleanup operation and erases and then rewrites the blocks and creates the blocks with all the deleted files. This achieves the goal of the user being able to use flash memory the same as they would use any other mass storage media without doing anything different.

These history paths comprise the file system operation and implementation.

- A File System Connector, whose job is to intercept the disk operations passed to MS-DOS by an application and translate them into generic file operations, passing them on to the system driver.
- A File System Driver, which accepts generic file operations passed to it from the File System Connector, implements the architecture and logic of the Microsoft Flash Filing System, and passes low-level commands such as read, write, copy, and erase to the device driver.
- A Device Driver, which accepts low-level commands from the File System Driver and interfaces to the host system hardware implementation.

The File System Connector performs a task analogous to a network interface for LAN (Local Area Network) systems and appears as a disk to MS-DOS. Any application that does not rely on a networked file system class of applications and utilities will not operate in this network. Specifically, those applications that issue the INT 13h disk BIOS I/O call, INT 21h DosRead, or INT 21h DosWrite calls will not work properly with the Flash Filing System, just as they would not work with a network LAN. The file system driver treats the flash media as a collection of large blocks all identical in size. Individual block statistics are kept within a variable-length structure at the top of

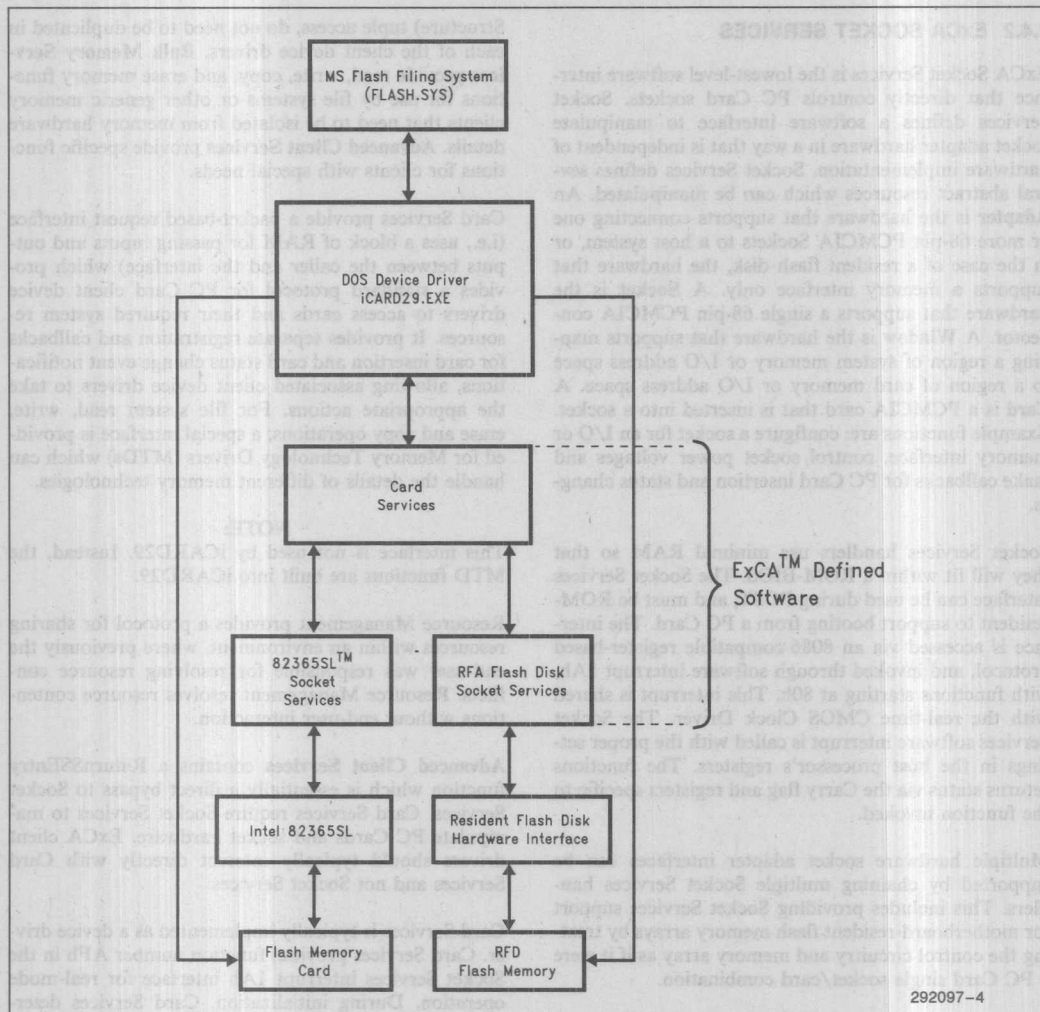


Figure 5. By using the same software architecture used for PCMCIA cards, less software duplication is present in systems containing both cards and Resident Flash Disks.

2.4.2 ExCA SOCKET SERVICES

ExCA Socket Services is the lowest-level software interface that directly controls PC Card sockets. Socket Services defines a software interface to manipulate socket adapter hardware in a way that is independent of hardware implementation. Socket Services defines several abstract resources which can be manipulated. An **Adapter** is the hardware that supports connecting one or more 68-pin PCMCIA Sockets to a host system, or in the case of a resident flash disk, the hardware that supports a memory interface only. A **Socket** is the hardware that supports a single 68-pin PCMCIA connector. A **Window** is the hardware that supports mapping a region of system memory or I/O address space to a region of card memory or I/O address space. A **Card** is a PCMCIA card that is inserted into a socket. Example functions are: configure a socket for an I/O or memory interface, control socket power voltages and make callbacks for PC Card insertion and status changes.

Socket Services handlers use minimal RAM so that they will fit within a ROM-BIOS. The Socket Services interface can be used during POST, and must be ROM-resident to support booting from a PC Card. The interface is accessed via an 8086-compatible register-based protocol, and invoked through software interrupt 1Ah, with functions starting at 80h. This interrupt is shared with the real-time CMOS Clock Driver. The Socket Services software interrupt is called with the proper settings in the host processor's registers. The functions returns status via the Carry flag and registers specific to the function invoked.

Multiple hardware socket adapter interfaces can be supported by chaining multiple Socket Services handlers. This includes providing Socket Services support for motherboard-resident flash memory arrays by treating the control circuitry and memory array as if it were a PC Card single socket/card combination.

2.4.3 ExCA CARD SERVICES

Card Services is the interface used to manipulate ExCA-related system resources. Card Services is subdivided into five functional categories: Client Services, Resource Management, Client Utilities, Bulk Memory Services, and Advanced Client Services. **Client Services** provide for client initialization and the callback registration of clients. **Resource Management** provides basic access to available system resources, combining knowledge of the current status of system resources with the underlying Socket Services adapter control functions. **Client Utilities** perform common tasks required by clients so that functions, such as CIS (Card Information

Structure) tuple access, do not need to be duplicated in each of the client device drivers. **Bulk Memory Services** provide read, write, copy, and erase memory functions for use by file systems or other generic memory clients that need to be isolated from memory hardware details. **Advanced Client Services** provide specific functions for clients with special needs.

Card Services provide a packet-based request interface (i.e., uses a block of RAM for passing inputs and outputs between the caller and the interface) which provides a standard protocol for PC Card client device drivers to access cards and their required system resources. It provides separate registration and callbacks for card insertion and card status change event notifications, allowing associated client device drivers to take the appropriate actions. For file system read, write, erase and copy operations, a special interface is provided for Memory Technology Drivers (MTDs) which can handle the details of different memory technologies.

NOTE:

This interface is not used by iCARD29. Instead, the MTD functions are built into iCARD29.

Resource Management provides a protocol for sharing resources within an environment, where previously the end-user was responsible for resolving resource conflicts. Resource Management resolves resource contentions without end-user interaction.

Advanced Client Services contains a ReturnSSEntry function which is essentially a direct bypass to Socket Services. Card Services require Socket Services to manipulate PC Cards and socket hardware. ExCA client drivers should typically interact directly with Card Services and not Socket Services.

Card Services is typically implemented as a device driver. Card Services provides function number AFh in the Socket Services interrupt 1Ah interface for real-mode operation. During initialization, Card Services determines the state of the host environment. This includes determining available system memory, available I/O ports, IRQ assignments, installed PC Cards and socket state. How this is performed is implementation specific.

2.4.4 iCARD29 FILE SYSTEM DRIVER

iCARD29.EXE is an Intel developed, low-level, flash memory driver for the MS Flash Filing System. It provides read, write, copy, and erase functions within the ExCA architecture and interfaces to a PCMCIA standard Card Services 2.0 (which provides proper resource arbitration). iCARD29 is completely independent of other peer level drivers and performs no direct calls to

Socket Services. Support for Intel's Series 1 and Series 2 flash memory cards and the RFA flash disk is provided. iCARD29 is also used to read ROM cards and read/write SRAM cards.

2.4.5 RFD SOCKET SERVICES

RFD "SocketServices" functions similar to ExCA SocketServices in that the software interface to manipulate socket adapter hardware is preserved, but an RFD SocketServices does not control any sockets or cards or respond to card removal and insertion events. RFD SocketServices allows a resident flash disk implementation, through chipset logic or external logic, to appear to ExCA software and the system as another Adapter using a single Window mechanism, accessing permanently installed flash memory on a motherboard. All the rest of ExCA SocketServices functions are kept as they relate to this definition. An example of a non-working RFD SocketServices function is using it to configure an I/O card. If requested to complete such an operation, RFA SocketServices will respond with "Function not supported."

For more information on the specific hardware design used for a Resident Flash Disk, refer to Section 3.5.

2.5 XIP Graphical Users Interface (GUI) Overview

Many GUIs exist today, but not all are configured to run in a minimized XIP mode for portables. Some designs may implement a simple DOS-based pen interface on top of an XIP DOS, like Communications Intelligence Corporation's PenDOS*, and add a single application like a forms recorder. Other designs may not use XIP DOS at all and the system design revolves around the XIP GUI requirements alone.

Microsoft leads the rest of the software industry in XIP GUI development, releasing the Windows 3.1 ROM Development Kit in September of 1992 and recently introducing the Modular Windows Development Kit in January of 1993. Both are XIP GUI implementations of the Windows GUI Operating System and are fully modularized for OEM configuration. Modularization assists OEMs by simplifying the streamlining of an O/S's suitability to task by allowing the OEM to choose which functions are important and required for a particular design and which functions can be left out. Benefits to the OEM are:

- Preserved API for using existing Windows applications or new application development
- Reduced development time and costs by using standard Windows application development tools and a wide choice of Windows software developers
- Ease of use for end-user

Modular Windows

Microsoft's Modular Windows Operating System uses a subset of the Windows 3.1 Operating System and includes extensions supporting TV-based multimedia players. Target market is home entertainment, but could easily be adapted for machine control on factory floors. The differences between Modular Windows and Windows 3.1 ROM are summarized below:

- Reduced support for the Windows 3.1 application programming interface (API)
- Reduced support for Windows 3.1 extension libraries
- New user-interface controls (instead of pull-down menus)
- New support for hand-control input devices

Software requirements are:

- MS DOS 5.0 in XIP form

System requirements are:

- 80286 or greater CPU
- 1 MB RAM minimum
- 1 MB of XIP memory (Flash, EPROM, or ROM)

Since the focus of this application note is personal computers, Sub-Notebooks and below, Modular Windows implementations will not be discussed. However, many of the principles of putting Modular Windows code in flash memory (memory maps, software tools, flash updatability, etc.) are the same here as the Windows 3.1 ROM example which is discussed later in this section.

Windows 3.1 ROM Version

Computers running Windows in ROM or XIP mode are very similar to standard PC running disk-based Windows. The only major exception is the presence of a large amount of XIP code storage in extended memory from which Windows executes, and a smaller amount of XIP code storage in adapter space. For the rest of this discussion, the Windows XIP code stored in extended memory is referred to as HIROM.BIN and the small amount of Windows XIP code stored in Real Mode space is referred to as LOROM.BIN.

Two modes of operation are possible for XIP Windows; Standard and Enhanced, just as on disk-based PCs. However, each require different system resources for the XIP Windows version.

For Standard Mode, Windows executes fully in ROM, leaving almost all the system RAM available for user programs. This means that all Windows "core code" including DOSX.EXE, USER.EXE, GDI, the Windows kernel and all drivers run from XIP storage space. Also, all shell programs, applets, fonts and other Windows resources are stored in and run from XIP storage space without being loaded into RAM.

Enhanced Mode Windows must execute partially from RAM. Enhanced Mode components such as WIN386.EXE and VxDs (virtual device drivers) must be loaded into RAM from some type of disk (a flash disk, ROM disk, or flash card) for execution, as their code writes back to their execution location from time to time and creates errors if loaded into XIP code storage such as flash memory. Components shared between modes, specifically USER.EXE, GDI, the kernel and drivers, continue to run from their stored locations in XIP address space.

Additionally, for either Standard or Enhanced modes, all Windows 3.1 features are supported in XIP Windows. The only limiting factors are the amounts of available RAM, XIP storage and, in the case of Enhanced mode, disk space.

Adding Applications

The Windows 3.1 ROM Development Kit (RDK) can add any Windows executable program or application into the main XIP binary image file HIROM.BIN. The application must conform to the XIP application requirements specified in the Windows 3.1 Technical Specification. Microsoft is capable of supplying XIP versions of Word for Windows, Excel, Microsoft Mail and Microsoft Works but must be developed between an OEM and Microsoft on a platform-by-platform basis.

Once the O/S and application functionality is determined and the build script CONTENTS.ROM edited, the RDK produces two binary images: the large HIROM.BIN Extended mode image containing O/S and application code, and the small Real mode LOROM.BIN image containing the listing of all the XIP code contained in HIROM.BIN. After both XIP Windows binary files are loaded into flash memory and the system is up and running, clicking on an application causes the XIP Windows kernel to search its internal listing (LOROM.BIN) for an XIP image module first. If the image is not found within the XIP listing, the Windows kernel then searches the file paths present at runtime to try and load the application from its current file directories. If the program is not found within

file directories, XIP Windows returns "Application missing, file not found." Additions of other extensions to the XIP image such as multimedia support and Pen Windows are added in a similar manner.

2.6 XIP GUI Implementation

As mentioned earlier, XIP Windows requires two modules; a small amount of XIP storage in device adapter space called LOROM.BIN, and a larger amount of XIP storage called HIROM.BIN in extended memory space. The LOROM.BIN file contains information about the modules loaded in the HIROM.BIN XIP (see Figure 6) image and must be accessed in real mode by three modules, WIN.COM, RSWAP.EXE, and WIN386.EXE. Additionally, portions of the DOS extender (DOSX.EXE) must be able to run in real mode and are therefore located in the LOROM.BIN XIP image. The information the three modules look for is contained in the ROM Table Of Contents (ROMTOC), also located in LOROM.BIN and contains general information about both XIP images (small and large), entry point addresses for initialization, and a list of the executables stored in the large XIP image.

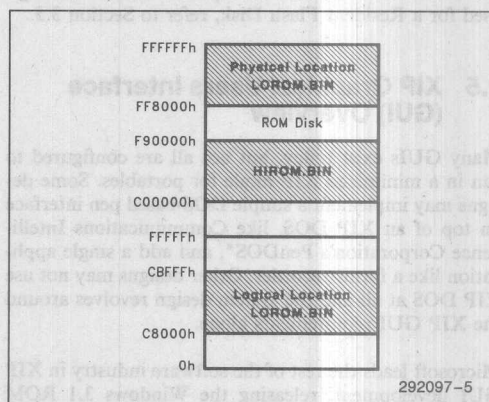


Figure 6. 16 MByte XIP GUI Memory Map

The extended memory XIP image contains the bulk of the system's code and data segments, EXE headers, and a prototype Local Descriptor Table (LDT) which is a data structure defining the addresses, sizes and types of segments used by 80286, Intel386™, or Intel486™ processors. For more information on a Windows 3.1 XIP-based system, please refer to the Microsoft Windows 3.1 ROM Development Kit.

■ 16MB or equivalent in XIP mode

- Software Utility to load both small and large XIP images into flash memory
- For Enhanced Mode disk space requirements, MS Flash Filing System 2.0 combined with ExCA software for a Resident flash disk.

System Requirements

- 80286 CPU or greater
- RAM: Minimum of 1 MB for Standard Mode or 2 MB for Enhanced mode
- XIP Store: Minimum of 2 MB for Standard Mode, 3+ MB for Enhanced mode
- Flash Disk Store: Zero for Standard Mode, 2 MB for Enhanced mode

For this particular design example, Enhanced mode functionality is used to show a full implementation of Windows. An RFD is used to load WIN386.EXE, VxDs and all the *.INI and *.GRP files. A complete listing of the CONTENTS.ROM file used by the Windows ROM Image Builder utility to create an XIP Windows system is shown in Appendix D. The Windows RDK Enhanced Full sample CONTENTS.ROM file is used as a template and edited to locate the LOROM.BIN file at C8000h and is a total of 16 KB. The HIROM.BIN file is located at C00000h and is a total of 3.6 MB.

2.7 Pen Extensions

Microsoft's Windows for Pen Computing is an extension to Microsoft Windows version 3.1 and has its own SDK. Pen Windows Extensions do not require any changes to existing Windows 3.1 applications and a se-

SDK provide for fast development of a pen-based system. In particular, the Wacom PL-100V pen tablet and VGA graphics card can be added to an ISA bus slot for early debug and pen software development.

All the Pen Windows extensions are directly executable within the HIROM XIP image and can be added to the CONTENTS.ROM listing.

3.0 HARDWARE

This section describes the general hardware requirements for an RFA design, then discusses a specific implementation using the SL architecture as an example (primarily due to the built-in ISA Sliding Window). Many of the concepts presented here can easily be incorporated into new chipsets designs to take advantage of the benefits flash memory brings to solid-state designs.

3.1 Resident Flash Disk Implementation

As stated in Section 2.3, the MS Flash Filing System requires a hardware mapping window in Real Mode address space. This window is similar in function to an EMS mapping window, but unlike EMS this window interfaces directly to flash memory. The SL Superset provides an example of just such a mapping, using in the ISA Sliding Window. The window is configured via the SL's ISAWINDOW register and has a fixed base address of D0000h and a fixed size of 64 KB. By setting the correct address in the register, the full 16 MB ISA address space is viewable in 64 KB increments. Additionally, the register defines access to the 16 MB flash disk address space made available by a separate flash disk Chipselect (see Figure 7).

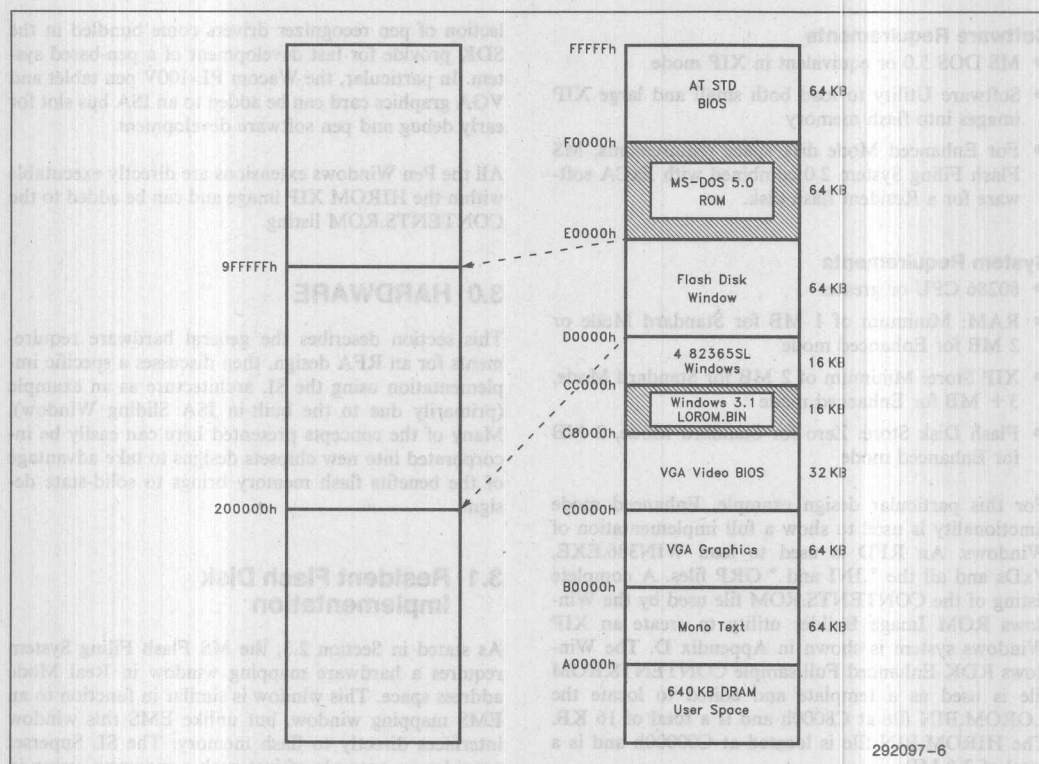


Figure 7. Flash Disk Mapping

Although the built-in functionality of the ISA Sliding Window is quite nice, the large 64 KB footprint of the window in Real Mode address space is difficult to work around. For a non-SL implementation, smaller, more flexible sliding windows are possible through external logic (FPGAs or EPLDs) or hopefully from future PC chipsets desiring to provide the capability the flash memory offers. The hardware requirements are simply:

1. A Window base address that appears in Real Mode memory somewhere in adapter space (C0000h to DFFFFh) that is also register definable.
2. A Window size that is also register definable for either 4 KB, 8 KB, 16 KB, or 32 KB.

These two options provide a level of flexibility for an OEM's system implementation and reduces the memory footprint in adapter space.

The flash disk address mapping in Figure 7 shows the flash disk and ISA bus address maps together. The SL's ISA Sliding window allows 64 KB blocks of the flash disk address space to be mapped into the Real mode area from D0000h to DFFFFh for access by the MS FlashFile System. For some ideas on how to implement an external logic flash disk implementation, see AP-343 "Solutions for High Density Applications Using Intel Flash Memory," order number 292079. The application note describes a complete design for an ISA Bus add-in card. A local bus design can be derived from the ISA Bus implementation or the SL implementation logic discussed in Section 3.6, Schematic Overview.

3.2 XIP DOS Implementation

As stated in Section 2.2 DOS in Flash Implementation, MS DOS 5.0 ROM Version is built assuming the E0000h segment location and also consists of a ROM Disk located in extended memory at F90000h. The only reason for the ROM disk location is to avoid 386SL co-

processor errata at the 8 MB location and window base address constraints, which must be divisible by the window size in use. In non-SL designs, the ROM Disk can be located anywhere above 1 MB in a NON-cacheable region. The design example uses Intel's 2 Mb, 28F002BX-T boot block flash memory. This device is organized with varying sized blocks; a 16 KB hardware lockable boot block, two 8 KB separately erasable Parameter Blocks, and separately erasable 96 KB and 128 KB main code blocks. To unlock and allow programming and erasure of the boot block, an additional 12V must be applied to the PWD# pin, thereby guaranteeing hardware protection. The benefit of using boot block architecture is that in the unlikely event that something happens during a BIOS code update, the system can recover using the kernel code in the boot block to initialize enough of the system to access a floppy drive or memory card socket to load in BIOS update code and a BIOS binary file.

Using a 2 Mb (256 KB) device enables the design to use a single memory chip for 4 separate code modules: the standard AT compatibility BIOS (64 KB), MS DOS 5.0 ROM (64 KB), Video BIOS (32 KB), and Power Management Code (32 KB). The sum of all 4 modules is greater than 128 KB and if mapped straight down from the top of the 1 MB address space, could cover both the BIOS and all of the adapter space. To avoid this conflict, some chipset designs physically position the BIOS function at the top of the 16 MB memory map, then use 64 KB mapping windows to position the correct block between E0000h and FFFFFh.

The SL Superset provides a second BIOS ROM chipselect enabling only one 128 KB portion to appear at E0000h to FFFFFh at a time. Additionally, the SL Superset provides an ISA Sliding Window which allowed windowed access into protected mode space. Figure 8 shows how each module is mapped into the boot block flash memory.

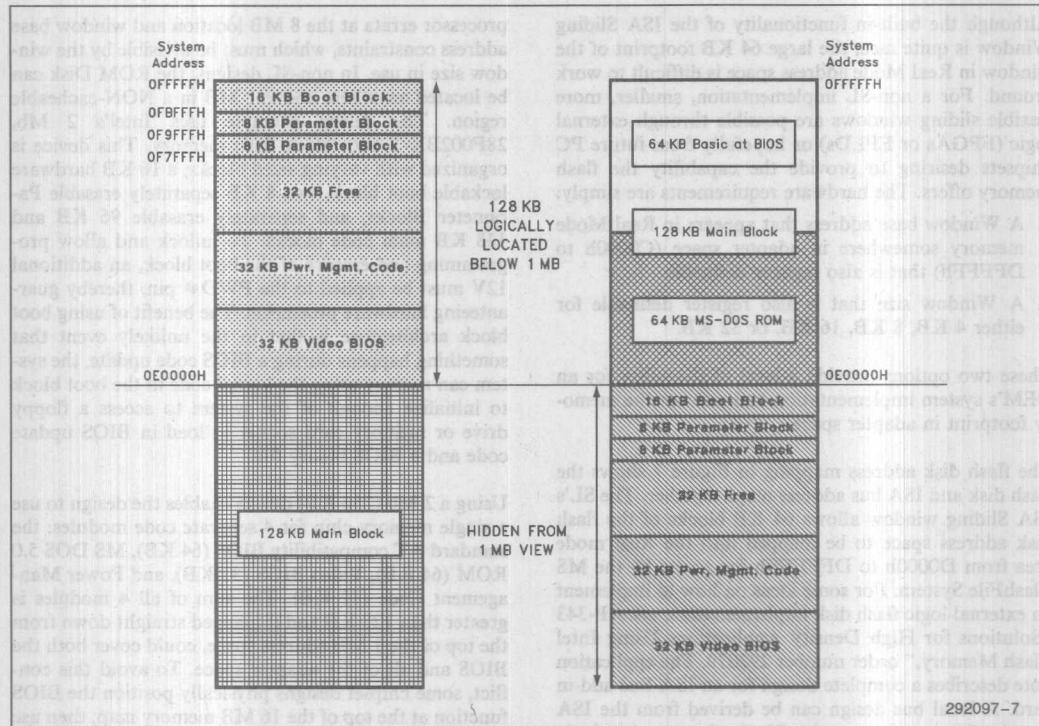


Figure 8. Boot Block Mapping

The boot block flash memory chip is physically located at the top of the 16 MB address space, but the SL Superset logically locates the top 128 KB of the chip into Real Mode address space just below 1 MB. Additionally, an exclusive OR gate is tied to the highest order address line, flipping the 2 Mb boot block chip at its mid point. This places the locked boot block just under the first 128 KB of the part, while the other 128 KB containing the AT System BIOS and MS DOS 5.0 ROM appears at the top of the device. This is done to position the boot block (which contains BIOS recovery code) out of the IBM BIOS compatibility table area, allowing access to the AT System BIOS and MS DOS ROM code.

When the system boots, the processor jumps to the AT BIOS location by default. As the system is booting, the BIOS enables the ISA Sliding Window to access the other 128 KB of boot block flash memory, and proceeds to copy the Power Management code to its special location within the System Management Mode space. Next, the BIOS copies the Video BIOS code into shadow memory at C0000h to C7FFFh and turns off the ISA Sliding Window. The BIOS then scans the MS DOS 5.0 ROM adapter code, and allows MS DOS ROM to hook INT 19h, and finishes the rest of its POST before turning control over to DOS by issuing INT 19h.

As stated earlier, Windows 3.1 ROM Version takes up about 3.6 MB of extended address space (HIROM.BIN) and 16 KB of Real mode adapter address space (LOROM.BIN). HIROM.BIN can be located anywhere above 1 MB and should probably be high enough above the DRAM address space to allow additional DRAM to be added to the system by the end-user. Ideally, the system design should be able to cache the area where the Windows HIROM XIP code is located. This allows the system to take advantage of XIP code locality since XIP code should produce a very high cache hit ratio. Since the 28F008SA Flash-File memory is a x8 device, x16 access is accomplished by pairing two devices for a LOW Byte HIGH Byte configuration. This creates an erasable block size of 128 KB.

The LOROM.BIN file has a couple of implementation options. One method is to use a spare block out of the extended memory XIP region. This method requires external logic to decode the specific adapter space address dedicated to the LOROM function, and generate a chipselect to the last block of flash memory. Since the LOROM.BIN file size is only 16 KB and the smallest erasable block size is 128 KB, 112 KB of the block is left unused. Given that the HIROM and LOROM files are updated together, the HIROM file could feasibly use the extra space if necessary.

Another option for the LOROM file is to use some free storage space within the 2 Mb boot block flash memory chip. The file could then be copied to the correct adapter space shadow RAM location at boot time. Copying the LOROM.BIN file can occur at the same time that Video BIOS and Power Management code are copied. This method provides update capability while reducing external logic requirements. The only hinge factor is getting the system's BIOS code to copy the file before or during POST.

Either option is possible. The choice is dependent on determining which is easier, modifying hardware or modifying a BIOS bootup process.

3.4 Chipset Considerations

Chipset designers interested in implementing an RFA-ready chipset should consider the following recommendations for RFA hardware requirements:

Flash BIOS > 128 KByte

- Minimally, include a Flip Bit on the highest order address line. This allows 128 KB pages to move in and out of the same 128 KB address space.

dow mapping mechanism similar to an EMS-like window. This window provides temporary Real mode access of the 2nd 128 KB of BIOS code storage.

RFD Functionality

- Include a built-in Flash disk Chipselect to minimize external logic
- Add an EMS-like mapping window
 - Register defined, base address between C0000h and DFFFFh
 - Register defined window size adjustable 4 KB, 8 KB, 16 KB, and 32 KB
 - Register address capability to 64 MB RFD

XIP Code Functionality

- Include a built-in XIP Flash Chipselect
- Add 2 register defined, cacheable windows
 - Adjustable base address anywhere above 1 MB
 - Size adjustable 2 MB, 4 MB, or 8 MB

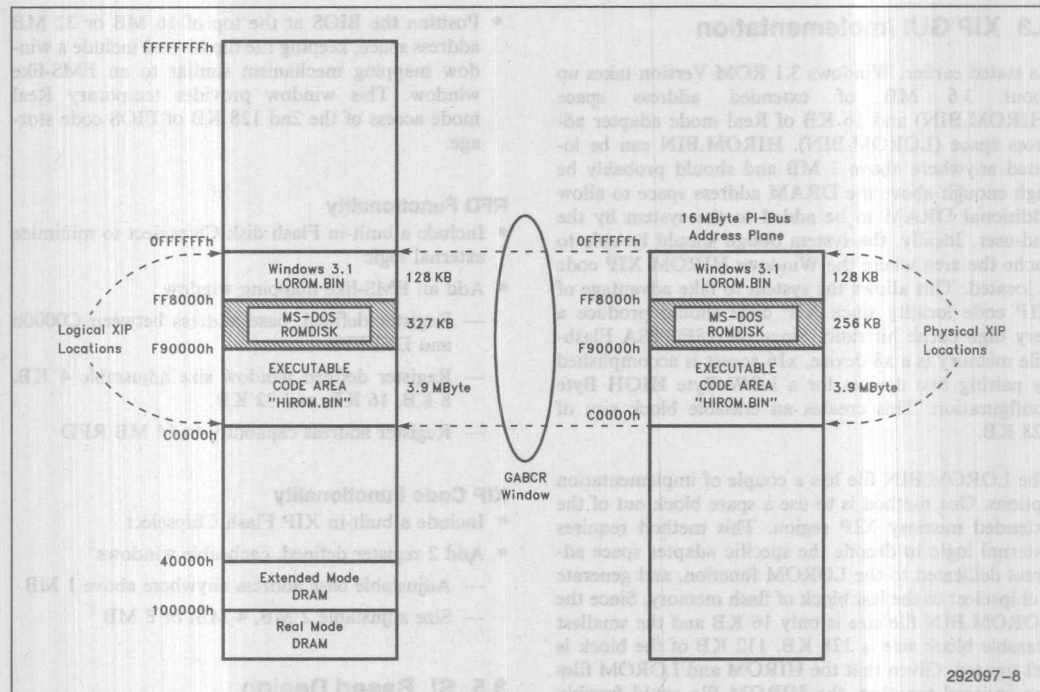
3.5 SL Based Design

Lacking an off-the-shelf, RFA-ready chipset and system, an SL Superset design is used as an example. To achieve as close to local bus access times as possible, the SL's Peripheral Interface (PI) Bus is used. The PI Bus is asynchronous and runs a CPU cycles plus one additional wait state.

To position the FlashFile memory on the PI Bus into the system's address map, a spare Video Window register (GABCR) is used (see Figure 9). For the design example, a base address of C00000h is used combined with a window size of 4 MB. The C00000h base address avoids some conflicts with 386SL systems at the 8 MB address location. Since the GABCR's window base address must be divisible by its size, the 12 MB address made the most sense (avoiding the 8 MB location) while being high enough to allow at least 8 MB of system RAM. For any other XIP system design, any base address should be usable. Although the GABCR register positions the FlashFile memory correctly and routes all system address map accesses to C00000h to FFFFFFFh to the PI Bus, by the fact that the GABCR is a graphics window, it is defined as a non-cacheable window.

- Include a cache enable/disable bit within the register

These requirements work for non-Intel architecture just as well as for the Intel architecture.



3.6 Schematic Overview

Appendix F contains schematics for the SL Superset example design. This section reviews the major portions of the schematic.

Even though the SL Superset provides a flash disk Window and Chipselect, two EPLDs are needed to decode the flash disk Chipselect from Video Chipselects, while the XIP flash memory required externally generated chipselects. Additional logic is also needed to decode the LOROM.BIN stub at CC000h and route the access to the highest block of XIP Memory.

The RFA example design uses both an Intel 85C090-15 and an Intel 5AC312-25 EPLD to provide board con-

trol logic. This logic generates flash memory chip enables, flash memory control signals, ROM Stub decoding, PI bus cycle decoding and bus cycle termination, and enables the mode registers. For details on the EPLD equations, see Appendix E.

85C090

The Intel 85C090 also controls the mix between how much flash memory is used for Resident Flash Disk Memory (FDM) and how much flash Disk Memory (RFD) and how much flash memory is used for XIP Memory. The EPLD description files in Appendix E show how to change the equations to obtain the desired mix needed for any particular OEM's implementation.

5AC312 Mode Registers

The 5AC312 provides two registers to control Vpp, power down, and the Ready Interrupt. The registers are written and read with PI Bus Memory cycles to the upper 64 KB of the Flash Disk Address Plane. This space is otherwise unused; no flash memory resides there. Address assignments in the flash disk plane are as follows:

- FF0000h Power Control Register (Wt/Rd)
- FF0002h Interrupt Control Register (Wt/Rd)
- FF0004h (Reserved)
- FF0006h Clear Flash Disk Memory READY Interrupt (Wt only)

The Clear Flash disk Memory READY Interrupt is not used with FlashFile System 1.0. It is intended to be used with the next file system from Microsoft. The above assignments alias every 8 Bytes (4 words) up to FFFFFEh.

The Power Control Register bits are defined as follows:

Bit	
0	Enable all Flash Memory Vpp (if jumper E9-E10 installed)
1	(Reserved; reads as a zero)
2	(Reserved; reads as a zero)
3	Power Down all flash memory
4-15	Undefined

The Interrupt Control Register bits are defined as follows:

Bit	
0	Enable Flash Memory READY Interrupt
1	Flash Memory READY Interrupt (Read only)
2	Flash Memory Any Zone Busy (Read only)
3	(Reserved; reads as zero)
4-15	Undefined

Reserved bits should be written as zeroes; they will read as zeroes. Undefined bits should be written as zeroes; they may read as either zeroes or ones.

Cold boot (RSTDRV) clears all defined bits in the register. Warm boot (CTRL-ALT-DEL) does not affect the contents of the register.

The Flash Memory READY interrupt is cleared by writing to address FF0006h in the flash disk Address plane. The data written is not interpreted; it should be all zeros.

A major power consumer on the RFA example design board is the 85C090 EPLD. This is due to using Turbo Mode to quickly decode the PI Bus signals and minimize RFA access time. Unfortunately, this causes a constant current draw of 160 mA. The EPLD's Normal mode cannot be used since there is no advance warning of an XIP flash memory access.

PRDY# Enable Jumper

The PI Bus signal PRDY# has a very weak pull-up resistor on the motherboard. Consequently, it has to be driven high (inactive) for a short time at the end of each cycle. Timing for this logic comes from the early taps on the delay line as follows:

100 Nanosecond delay line
(SXTTLDM-121)E4-E5

200 Nanosecond delay line
(SXTTLDM-125)E4-E3

Power Control

The purpose of the SMOUT4 signal was to allow the RFA Daughter board to perform a local standby, powering down various devices on the RFA Daughter board. However, since the EPLD must remain powered on until Global suspend, and consumes a high amount of current compared to the FlashFile power consumption, local standby was not implemented.

4.0 SOFTWARE UTILITIES

A few software utilities need to be created to load the binaries created for XIP code implementations.

4.1 RFA Diagnostic

It is highly recommended that system designers develop simple diagnostic tools to test the hardware at a very low level (i.e., write byte, write word, read word, erase block). Such a tool proves invaluable when debugging new hardware and software designs and resolving hardware and software conflicts.

4.2 RFA Binary Loader

The RFA Binary files HIROM.BIN, LOROM.BIN, and ROM Disk must all be loaded into Protected Mode from Real Mode. This can be accomplished a number of ways:

1. Use a DOS Extender. This provides a quick method to create a utility using the tools a DOS extender provides. However, licensing may prove to be difficult or expensive.
2. Using BIOS-based extended memory calls. These actually worked quite nicely and reliably, but proved to be slow and doubled the time to erase and program HIROM.BIN.
3. Putting the processor in flat mode. This method allows for fast, direct access to the extended memory, but cannot be done under windows.

For lab testing, our binary loader used a simple file name plus command line parameters. A system for end-users would need a more elaborate user interface to guide them through the software update. Some basic software requirements outside of the basic file read/write capabilities and command line parsing for the flash memory Binary Loader are:

- Incorporate basic flash memory program and erasure commands. These software drivers are available for both ASM86 and "C" in Application Note-360 "28F008SA" Software Drivers, order number 292095-002. This application note addresses things like read device identifier, V_{pp} ramp time, x8 and x16 parallel programming and block erasure by providing proven, tested routines for each.

- Choose a method of Protected Mode access that makes the most sense for you.
- Allow for specific base addresses to be entered by the user, while within the program automatically determining what block and the number of blocks to be erased from the base address and binary file size.

5.0 SUMMARY

This application note discussed a new system architecture based on solid-state software and hardware design concepts. This new architecture is based on using flash memory for the following; BIOS + DOS code storage, a nonvolatile RAM disk or RFD, and as XIP GUI code storage or RFX. Specific flash memory component and PCMCIA card information is found in their respective datasheets. Contact your local Intel or distribution sales office for more information or to obtain assistance in evaluating Boot Block or FlashFile memory components, as well as Intel's product line of PCMCIA Flash Memory Cards.

APPENDIX A ADDITIONAL PUBLICATIONS

- Intel "28F008SA Hardware Interfacing" Application Note
Order number 292094
- Intel "28F008SA Software Drivers" Application Note
Order number 292095
- Intel "Power Supply Solutions for Flash Memory" Application Note
Order number 292092
- Intel 85C090 24-Macrocell CHMOS EPLD
Order number 290247
- Intel 5AC312 12-Macrocell CHMOS EPLD
Order number 290247

292097-9

APPENDIX A ADDITIONAL PUBLICATIONS

- > Intel "286/386 Hardware Interfacing" Application Note
Order number 282094
- > Intel "286/386 Software Drivers" Application Note
Order number 282092
- > Intel "Power Supply Solutions for Flash Memory" Application Note
Order number 282093
- > Intel 85C099 24-Macrocell CMOS EPLD
Order number 290347
- > Intel 3A017 12-Macrocell CMOS EPLD
Order number 290347

APPENDIX B MS DOS ROM IMAGE DESCRIPTION

```
#####
#           RFA ROM DOS Description File                               #
# ROM image description file for 64K of ROM space at                   #
#   E0000-EFFFF and one 256K ROMDISK module at F90000                 #
#####
# Actual file sizes created: Three 32K modules
```

```
ROM1=Int 19 hook and Resident DOS Code
ROM1SIZE=8000
ROM1MAX=7FFF
ROM1TYPE=SEG
ROM1ADDR=E000
ROM1CHKSUM=YES
ROM1NUMBLOCKS=40
ROM1FILES=..\romhead\romboot.bin ..\dos\resdos.16
```

```
ROM2=COMMAND ROM Hdr Res. BIOS Code Bootstrap loader Resident Command Code
ROM2SIZE=8000
ROM2MAX=7FFF
ROM2TYPE=SEG
ROM2ADDR=E800
ROM2CHKSUM=YES
```

292097-10

```
ROM2NUMBLOCKS=40
ROM2FILES=..\cmd\command\romhead.bin ..\bios\resbio.16 \
..\romload\romload.sys ..\cmd\command\rescom.16 ..\dos\romdos.sys
```

```
ROM3=Command interpreter
ROM3SIZE=10000
ROM3MAX=FFFF
ROM3TYPE=BASE
ROM3ADDR=F90000
ROM3FILES=..\cmd\command\command.16 ..\bios\rombio.sys
```

292097-11

APPENDIX C

ROMDISK CONFIG.SYS AND AUTOEXEC.BAT

AUTOEXEC.BAT

```
@echo off
prompt $P$G
set path=c:\;d:\rdk10\build\disk;d:\utils;d:\diags;
c:\doskey
doskey d=dir $1 $2
d:\rdk10\build\disk\smartdrv.exe 1024 1024
set TEMP=d:\
ver /r
echo "MS DOS ROM, Card & Flash Disk SS 2.0, MS FlashFile System 2,"
echo "and Windows 3.1 ROM "
```

CONFIG.SYS

```
DEVICE=c:\HIMEM.SYS
break=on
buffers=40
files=40
lastdrive=H
DOS=HIGH,UMB
REM *****
REM FlashFile System Drivers
Device=C:\rfaslss.sys
Device=C:\ss365sl.exe
Device=C:\cs.exe
Device=C:\rtinit.exe
Device=C:\icard29.exe
Device=C:\ms-flash.sys
REM *****
```


APPENDIX D WINDOWS 3.1 CONTENTS.ROM

NOTE: Semi-colons denote a commented-out line which is NOT added to the HIROM Binary file.

```
*****
; Windows 3.1 ROM Development Kit (RDK) 1.0
; Sample ROM Description File
; Copyright Microsoft Corporation, 1992
;
; Enhanced Mode, Full
; 10/20/92 Removed TT Fonts
```

ROMS

```
; Specifies length of ROMs and the linear addresses
; at which they are to appear.
;
; Name Address Length (max)
; -----
LoROM 0c8000 004000 ; 16k
HiROM C00000 3FFC00 ; 4 Mb
```

TABLES

; Specifies information for tables to reside in ROM.

```
ROMTOC 100 LoROM ; ROMTOC entries
NUMFILENT 14 ; FILES entries
LDT 1024 HiROM 256 ; Local Descriptor Table
; WINFLAGS 13 ; 286 version; Value is in HEX
; WINFLAGS 15 ; 386 version; Value is in HEX
; SYSDIR system ; Windows directory on disk (optional)
ROMVERSION 1000 ; 1 = masked ROM, 000 = OEM version
```

MODULES

; Specifies modules to be loaded into ROM.

; Format is as follows:

Module	SEG File	ROM	Flags	Comments
Kernel				

; Kernel

```
DOSX.EXE %ROMFILES%\dosx.exe LoROM NOEXEHDR ; Std mode MS DOS extender
SEG 2 HiROM ; DXDGROUP - Copy from ext with INT 15h
```

292097-13

```

SEG 3                HiROM                ; DXPMCODE
; KERNEL.EXE %ROMFILES%krm1286.exe HiROM    ; 286 kernel
KERNEL.EXE %ROMFILES%krm1386.exe HiROM    ; 386 kernel (ROM version)

```

; Drivers (Replaceable) -----

```

SYSTEM.DRV %ROMFILES%system.drv HiROM      ; System
KEYBOARD.DRV %ROMFILES%keyboard.drv HiROM  ; Keyboard
SEG 10 COMP                                ; Do not remove!

```

; Display

```

; VGAROM2.DRV %ROMFILES%vgarom2.drv HiROM  ; 286 VGA
VGAROM3.DRV %ROMFILES%vgarom3.drv HiROM    ; 386 VGA
; SVGAR2.DRV %ROMFILES%svgar2.drv HiROM    ; 286 SuperVGA
; SVGAR3.DRV %ROMFILES%svgar3.drv HiROM    ; 386 SuperVGA

```

```

MOUSE.DRV %ROMFILES%mouse.drv HiROM        ; Mouse
SEG 2 RAM COMP NORELOC                      ; Do not remove!
; NOMOUSE.DRV %ROMFILES%nomouse.drv HiROM   ; No mouse

```

```

COMM.DRV %ROMFILES%comm.drv HiROM          ; COM, LPT
SEG 2 RAM COMP NORELOC                      ; Do not remove!
SEG 3 COMP                                ; Do not remove!

```

```

MMSOUND.DRV %ROMFILES%mmsound.drv HiROM    ; Sound

```

; Core -----

```

GDI.EXE %ROMFILES%gdi.exe HiROM            ; ROM version
USER.EXE %ROMFILES%user.exe HiROM          ; ROM version
SEG 3 RAM COMP NORELOC                      ; Do not remove!

```

; Non-Replaceable System DLLs -----

```

SHELL.DLL %ROMFILES%shell.dll HiROM        ; Shell APIs
LZEXPAND.DLL %ROMFILES%lzexpand.dll HiROM   ; Expansion
WIN87EM.DLL %ROMFILES%win87em.dll HiROM    ; Math emulator

```

; Replaceable System DLLs -----

```

COMMDLG.DLL %ROMFILES%commdlg.dll HiROM STUB ; Common dialogs
OLECLI.DLL %ROMFILES%olecli.dll HiROM STUB   ; OLE Client
OLESVR.DLL %ROMFILES%olesvr.dll HiROM STUB   ; OLE Server
TOOLHELP.DLL %ROMFILES%toolhelp.dll HiROM STUB ; Tool Help DLL
DDEML.DLL %ROMFILES%ddeml.dll HiROM STUB     ; DDE
VER.DLL %ROMFILES%ver.dll HiROM STUB         ; Version DLL

```

; Multimedia Extensions -----

```

MMSYSTEM.DLL %ROMFILES%mmsystem.dll HiROM STUB ; Multimedia

```

SND.CPL %RETAIL%snd.cpl HiROM ; Sound icon
MPLAYER.EXE %RETAIL%mpplayer.exe HiROM ; Media Player
SOUNDREC.EXE %RETAIL%soundrec.exe HiROM ; Sound Recorder

; Advanced Power Management (APM) -----

; POWER.DRV %RETAIL%power.drv HiROM ; APM driver

; Shell Programs -----

PROGMAN.EXE %RETAIL%progman.exe HiROM ; Program Mgr
WINFILE.EXE %RETAIL%winfile.exe HiROM ; File Manager
TASKMAN.EXE %RETAIL%taskman.exe HiROM ; Task Manager
WINHELP.EXE %RETAIL%winhelp.exe HiROM ; Windows Help
WINTUTOR.EXE %RETAIL%wintutor.exe HiROM ; Tutorial

; Control Panel -----

DRIVERS.CPL %RETAIL%drivers.cpl HiROM ; Drivers icon
MAIN.CPL %ROMFILES%main.cpl HiROM ; Main icons
CONTROL.EXE %RETAIL%control.exe HiROM ; Control Panel

; Printing Support -----

; PRINTMAN.EXE %RETAIL%printman.exe HiROM ; Print Mgr
; UNIDRV.DLL %ROMFILES%unidrv.dll HiROM ; Uni driver
; DMCOLOR.DLL %ROMFILES%dmcolor.dll HiROM ; Uni driver

; System Fonts -----

VGASYS.FON %RETAIL%vgasys.fon HiROM ; System (VGA)
VGAFIX.FON %RETAIL%vgafix.fon HiROM ; Fixed pitch
VGAOEM.FON %RETAIL%vgaoem.fon HiROM ; OEM

; Bitmap Fonts -----

COURE.FON %RETAIL%coure.fon HiROM ; Courier (VGA)
SERIFE.FON %RETAIL%serife.fon HiROM ; MS Serif (VGA)
SMALLE.FON %RETAIL%smalle.fon HiROM ; Small (VGA)
SSERIFE.FON %RETAIL%sserife.fon HiROM ; MS Sans Serif (VGA)
SYMBOLE.FON %RETAIL%symbole.fon HiROM ; Symbol (VGA)

; Plotter Fonts -----

MODERN.FON %RETAIL%modern.fon HiROM ; Modern
ROMAN.FON %RETAIL%roman.fon HiROM ; Roman
SCRIPT.FON %RETAIL%script.fon HiROM ; Script

; TrueType Fonts -----

```

ARIAL.FOT %RETAIL%arial.fot HiROM ; Arial
ARIALBD.FOT %RETAIL%arialbd.fot HiROM ; Arial Bold
ARIALBI.FOT %RETAIL%arialbi.fot HiROM ; Arial Bold Italic
ARIALI.FOT %RETAIL%ariali.fot HiROM ; Arial Italic
; COUR.FOT %RETAIL%cour.fot HiROM ; Courier New
; COURBD.FOT %RETAIL%courbd.fot HiROM ; Courier New Bold
; COURBI.FOT %RETAIL%courbi.fot HiROM ; Courier New Bold Italic
; COURI.FOT %RETAIL%couri.fot HiROM ; Courier New Italic
; SYMBOL.FOT %RETAIL%symbol.fot HiROM ; Symbol
; TIMES.FOT %RETAIL%times.fot HiROM ; Times New Roman
; TIMESBD.FOT %RETAIL%timesbd.fot HiROM ; Times New Roman Bold
; TIMESBI.FOT %RETAIL%timesbi.fot HiROM ; Times New Roman Bold Italic
; TIMESI.FOT %RETAIL%timesi.fot HiROM ; Times New Roman Italic
WINGDING.FOT %RETAIL%wingding.fot HiROM ; WingDings

; MS DOS App Support -----

VGA.3GR %RETAIL%vga.3gr HiROM ; Enh mode grabber
WINOLDAP.MOD %RETAIL%winoldap.mod HiROM ; Std mode MS DOS app support
SEG 2 RAM COMP NORELOC ; Do not remove!
WINOA386.MOD %RETAIL%winoa386.mod HiROM ; Enh mode MS DOS app support
SEG 1 RAM COMP NORELOC ; Do not remove!
SEG 2 RAM COMP NORELOC ; Do not remove!
SEG 5 RAM COMP NORELOC ; Do not remove!

DOSAPP.FON %RETAIL%dosapp.fon HiROM ; MS DOS app window fonts
EGA80WOA.FON %RETAIL%ega80woa.fon HiROM
EGA40WOA.FON %RETAIL%ega40woa.fon HiROM
CGA80WOA.FON %RETAIL%ega80woa.fon HiROM
CGA40WOA.FON %RETAIL%ega40woa.fon HiROM

; Applets -----

CALC.EXE %RETAIL%calc.exe HiROM ; Calculator
CALENDAR.EXE %RETAIL%calendar.exe HiROM ; Calendar
CARDFILE.EXE %RETAIL%cardfile.exe HiROM ; Cardfile
CHARMAP.EXE %RETAIL%charmap.exe HiROM ; Character Map
CLIPBRD.EXE %RETAIL%clipbrd.exe HiROM ; Clipboard Viewer
CLOCK.EXE %RETAIL%clock.exe HiROM ; Clock
NOTEPAD.EXE %RETAIL%notepad.exe HiROM ; Notepad applet
PACKAGER.EXE %RETAIL%packager.exe HiROM ; Packager applet
PBRUSH.DLL %RETAIL%pbrush.dll HiROM ; for Paintbrush
PBRUSH.EXE %RETAIL%pbrush.exe HiROM ; Paintbrush
PIFEDIT.EXE %RETAIL%pifedit.exe HiROM ; PIF Editor
RECORDER.DLL %RETAIL%recorder.dll HiROM ; for RECORDER.EXE
RECORDER.EXE %RETAIL%recorder.exe HiROM ; Recorder
SOL.EXE %RETAIL%sol.exe HiROM ; Solitaire
; TERMINAL.EXE %RETAIL%terminal.exe HiROM ; Terminal

```



```
WINMINE.EXE %RETAIL%winmine.exe HiROM ; WinMine
WRITE.EXE %RETAIL%write.exe HiROM ; Write
```

; Applications -----

FILES

; Specifies optional files to be installed into ROM.
; TrueType TTF font files are specified in this section.

```
; ROM Name Path ROM
; -----
```

; TrueType Data -----

```
ARIAL.TTF %RETAIL%arial.ttf HiROM ; Arial
ARIALBD.TTF %RETAIL%arialbd.ttf HiROM ; Arial Bold
ARIALBI.TTF %RETAIL%arialbi.ttf HiROM ; Arial Bold Italic
ARIALI.TTF %RETAIL%ariali.ttf HiROM ; Arial Italic
; COUR.TTF %RETAIL%cour.ttf HiROM ; Courier New
; COURBD.TTF %RETAIL%courbd.ttf HiROM ; Courier New Bold
; COURBI.TTF %RETAIL%courbi.ttf HiROM ; Courier New Bold Italic
; COURI.TTF %RETAIL%couri.ttf HiROM ; Courier New Italic
; SYMBOL.TTF %RETAIL%symbol.ttf HiROM ; Symbol
; TIMES.TTF %RETAIL%times.ttf HiROM ; Times New Roman
; TIMESBD.TTF %RETAIL%timesbd.ttf HiROM ; Times New Roman Bold
; TIMESBI.TTF %RETAIL%timesbi.ttf HiROM ; Times New Roman Bold Italic
; TIMESI.TTF %RETAIL%timesi.ttf HiROM ; Times New Roman Italic
WINGDING.TTF %RETAIL%wingding.ttf HiROM ; WingDings
```

Winning
Win

WINNING: RETAIL:winning.exe HROM
WIN: RETAIL:win.exe HROM

Application:

FILES

Specifies optional files to be installed into ROM.
TrueType font files are specified in this section.

ROM Name Path ROM

TrueType Data

Arial
Arial Bold
Arial Bold Italic
Arial Italic
Courier New
Courier New Bold
Courier New Bold Italic
Courier New Italic
Symbol
Times New Roman
Times New Roman Bold
Times New Roman Bold Italic
Times New Roman Italic
Wingdings

ARIAL.TTF RETAIL:arial.ttf HROM
ARIALB.TTF RETAIL:arialb.ttf HROM
ARIALBI.TTF RETAIL:arialbi.ttf HROM
ARIALI.TTF RETAIL:ariali.ttf HROM
COURT.TTF RETAIL:court.ttf HROM
COURBD.TTF RETAIL:courbd.ttf HROM
COURBIT.TTF RETAIL:courbit.ttf HROM
COURIT.TTF RETAIL:courit.ttf HROM
SYMBOL.TTF RETAIL:symbol.ttf HROM
TIMES.TTF RETAIL:times.ttf HROM
TIMESB.TTF RETAIL:timesb.ttf HROM
TIMESBI.TTF RETAIL:timesbi.ttf HROM
TIMESI.TTF RETAIL:timesi.ttf HROM
WINGDING.TTF RETAIL:wingding.ttf HROM

200007-11

APPENDIX E EPLD EQUATIONS

Included below are the equations for both the 85C090 and the 5AC312 EPLDs. See the RFA Schematic Diskette for disk-readable *.ADF and *.JED files.

85C090 EPLD

CN_SPC_A.ADF
85C090

OPTIONS: TURBO=ON
PART: N85C090 % PLCC %

%

292097-18

3

This EPLD provides the main control logic of the Resident Flash Array Evaluation board. The following is included:

Flash Memory chip enables
 Modes Register chip enable (RS_CE/FLASH_CES)
 ROM Stub control logic
 74FCT623 Data Buffer Control
 74FCT373 Address Latches Gate control (HOLD#)
 WT, WT#, and RD# generation

CN_REV_A - 04/06/92 Initial release

CN_SPC_A - 04/08/92 Reconfigured Flash Array to provide 8 Meg
 of Flash Disk and 4 Meg of Executable Memory.

CN_SPC_B - 04/29/92 Update to allow byte writes.

CN_REV_B - 06/01/92 Added generalize configuration scheme.

%

INPUTS:

DL_OUT@2 % Delay line out (high edge sets PRDY f/t) %
 PM@4 % PI bus Memory or IO# %
 PCMD#@14 % PI bus Command %
 PSTART_DLY#@42 % Delayed PSTART# signal (used to create HOLD#) %
 A14@5 % PI/ISA bus Latched Addresses %
 A15@19
 A16@20
 LA17@21 % PI/ISA bus Unlatched Addresses %
 LA18@26
 LA19@25
 LA20@27
 LA21@28
 LA22@29
 LA23@30
 MRDC#@3 % ISA bus Memory Read (All 16MByte) %
 VGACS#@13 % VGA Chip Select %
 FLASHDCS#@12 % Flash Disk Chip Select %
 L_PW@41 % PI bus PW/R# (latched) %
 EN_PRDY@40 % Enable PRDY# driver %
 SA00@43 % Latched address bit 00 %
 SBHE_L#@37 % Latched System Byte High Enable %

OUTPUTS:

FLASH_CE0#@15 % Flash Memory Zone pairs Chip Selects %
 FLASH_CE1#@16
 FLASH_CE2#@35

292097-19

FLASH_CE3#@34
 FLASH_CE4#@33
 FLASH_CE5#@32
 RS_CE@31 % ROM Stub Chip Select
 RD#@6 % Read Strobe
 WT_L#@10 % Write Strobe, Lower Byte
 WT_U#@8 % Write Strobe, Upper Byte
 WT@36 % Write Strobe (Active High for 74F623 enable) %
 HOLD#@7 % Hold Address Latches (close gate) %
 SA16@11 % HIGH for Sel_RS, else 3-S
 DL_IN@38 % Delay line driver
 PRDY#@9 % PI bus signal
 PRDY@18 % D-type f/f

NETWORK:

PM = INP(PM)
 nPCMD = INP(PCMD#)
 nPSTART_DLY = INP(PSTART_DLY#)
 A14 = INP(A14)
 A15 = INP(A15)
 A16 = INP(A16)
 LA17 = INP(LA17)
 LA18 = INP(LA18)
 LA19 = INP(LA19)
 LA20 = INP(LA20)
 LA21 = INP(LA21)
 LA22 = INP(LA22)
 LA23 = INP(LA23)
 nMRDC = INP(MRDC#)
 nVGACS = INP(VGACS#)
 nFLASHDCS = INP(FLASHDCS#)
 L_PW = INP(L_PW)
 EN_PRDY = INP(EN_PRDY)
 DL_OUT = INP(DL_OUT)
 SA00 = INP(SA00)
 nSBHE_L = INP(SBHE_L#)
 FLASH_CE0#,nFLASH_CE0 = COIF(_FLASH_CE0,VCC)
 FLASH_CE1#,nFLASH_CE1 = COIF(_FLASH_CE1,VCC)
 FLASH_CE2#,nFLASH_CE2 = COIF(_FLASH_CE2,VCC)
 FLASH_CE3#,nFLASH_CE3 = COIF(_FLASH_CE3,VCC)
 FLASH_CE4#,nFLASH_CE4 = COIF(_FLASH_CE4,VCC)
 FLASH_CE5#,nFLASH_CE5 = COIF(_FLASH_CE5,VCC)
 RS_CE,RS_Cef = COIF(iRS_CE,VCC)
 RD# = CONF(_RD,VCC)
 WT_L# = CONF(_WT_L,VCC)
 WT_U# = CONF(_WT_U,VCC)
 WT = CONF(iWT,VCC)

HOLD# = CONF(HOLD,VCC)
 DL_IN,DL_INF = COIF(iDL_IN,VCC)
 PRDY,PRDYf = RORF(VCC,DL_OUT,DL_IN,GND,VCC)
 PRDY# = CONF(PRDY,EN_PRDY)
 SA16 = CONF(VCC,Rd_RS)

EQUATIONS:

% Internal Declarations %

PCMD = nPCMD';
 PSTART_DLY = nPSTART_DLY';
 MRDC = nMRDC';
 VGACS = nVGACS';
 FLASHDCS = nFLASHDCS';
 SBHE_L = nSBHE_L';

FLASH_CE0 = nFLASH_CE0';
 FLASH_CE1 = nFLASH_CE1';
 FLASH_CE2 = nFLASH_CE2';
 FLASH_CE3 = nFLASH_CE3';
 FLASH_CE4 = nFLASH_CE4';
 FLASH_CE5 = nFLASH_CE5';
 _DL_IN = DL_INF';
 _PRDY = PRDYf';

Hold = PSTART_DLY
 + PCMD
 + MRDC;

Open = /Hold;
 Closed = /Open;

Sel_FDM = FLASHDCS; % Flash Disk Memory (mem cycles only) %

Sel_XIP = /FLASHDCS * /VGACS * PM; % Execute-in-place Memory %

% ROM Stub, 0C8000h through 0CBFFeh %

Sel_RS = /FLASHDCS * /VGACS * /LA23 * /LA22 * /LA21
 * /LA20 * LA19 * LA18 * /LA17 * /A16 * A15 * /A14;

Sel_Modes = Sel_FDM * LA23 * LA22 * LA21 * LA20 % Modes Reg %
 * LA19 * LA18 * LA17 * A16;

Modes_CE = RS_Cef * /FLASH_CE5;

Rd_RS = RS_Cef * FLASH_CE5;

Yes = VCC;

No = GND;

% Zone pairs 0 and 1 are unconditionally assigned to the Flash Disk. Zone pair 5 is unconditionally assigned to the Executable Memory. Zone pairs 2, 3, and 4 are assigned with the three equations below by selecting either Yes or No for the three Flash Disk assignments. Assignments must be in order; 3 can be assigned to the Flash Disk only if 2 is, and 4 can be assigned to the Flash Disk only if 2 and 3 is. Zone pairs not assigned to the Flash Disk are automatically assigned to Executable Memory. %

% Demo Configuration

Z2eqFD = No;

Z3eqFD = No;

Z4eqFD = No;

%

% Customer Configuration %

Z2eqFD = Yes;

Z3eqFD = Yes;

Z4eqFD = No;

Z2eqXIP8 = /Z2eqFD;

Z3eqXIP8 = /Z2eqXIP8 * /Z3eqFD;

Z3eqXIPA = Z2eqXIP8 * /Z3eqFD;

Z4eqXIP8 = /Z2eqXIP8 * /Z3eqXIP8 * /Z4eqFD;

Z4eqXIPA = Z3eqXIP8 * /Z4eqFD;

Z4eqXIPC = Z3eqXIPA * /Z4eqFD;

Z5eqXIP8 = /Z2eqXIP8 * /Z3eqXIP8 * /Z4eqXIP8;

Z5eqXIPA = Z4eqXIP8;

Z5eqXIPC = Z4eqXIPA;

Z5eqXIP8 = Z4eqXIPC;

% Outputs %

% Zone pairs 0 and 1 are unconditionally assigned to Flash Disk. %

_FLASH_CE0' = Open * Sel_FDM * /LA23 * /LA22 * /LA21
+ Closed * FLASH_CE0;

_FLASH_CE1' = Open * Sel_FDM * /LA23 * /LA22 * LA21
+ Closed * FLASH_CE1;

% Zone pairs 2, 3, and 4 are conditionally assigned to either Flash Disk or Execute-in-Place memory. XIP starts at 800000h to avoid conflict with ROM Stub. %

```
_FLASH_CE2' = Open * Sel_FDM * /LA23 * LA22 * /LA21 * Z2eqFD
+ Open * Sel_XIP * LA23 * /LA22 * /LA21 * Z2eqXIP8
+ Closed * FLASH_CE2;
```

```
_FLASH_CE3' = Open * Sel_FDM * /LA23 * LA22 * LA21 * Z3eqFD
+ Open * Sel_XIP * LA23 * /LA22 * /LA21 * Z3eqXIP8
+ Open * Sel_XIP * LA23 * /LA22 * LA21 * Z3eqXIPA
+ Closed * FLASH_CE3;
```

```
_FLASH_CE4' = Open * Sel_FDM * LA23 * /LA22 * /LA21 * Z4eqFD
+ Open * Sel_XIP * LA23 * /LA22 * /LA21 * Z4eqXIP8
+ Open * Sel_XIP * LA23 * /LA22 * LA21 * Z4eqXIPA
+ Open * Sel_XIP * LA23 * LA22 * /LA21 * Z4eqXIPC
+ Closed * FLASH_CE4;
```

% Zone pair 5 is unconditionally assigned to Execute-in-Place.
Zone pair 5 is also enabled if Rom Stub is selected. %

```
_FLASH_CE5' = Open * Sel_XIP * LA23 * /LA22 * /LA21 * Z5eqXIP8
+ Open * Sel_XIP * LA23 * /LA22 * LA21 * Z5eqXIPA
+ Open * Sel_XIP * LA23 * LA22 * /LA21 * Z5eqXIPC
+ Open * Sel_XIP * LA23 * LA22 * LA21 * Z5eqXIPE
+ Open * Sel_RS
+ Closed * FLASH_CE5;
```

% RS_CE forces Flash Addresses 15 through 20 high. This, along
with the activation of FLASH_CE5#, cause ROM Stub accesses to
go to the upper 16 KBytes of Zone pair 5. %

```
iRS_CE = Open * Sel_RS
+ Open * Sel_Modes
+ Closed * RS_CEf;
```

```
_RD' = PCMD * FLASH_CE0 * /L_PW
+ PCMD * FLASH_CE1 * /L_PW
+ PCMD * FLASH_CE2 * /L_PW
+ PCMD * FLASH_CE3 * /L_PW
+ PCMD * FLASH_CE4 * /L_PW
+ PCMD * FLASH_CE5 * /L_PW
+ PCMD * Modes_CE * /L_PW
+ MRDC * Rd_RS;
```

```
_WT_U = PCMD * FLASH_CE0 * L_PW * SBHE_L
+ PCMD * FLASH_CE1 * L_PW * SBHE_L
+ PCMD * FLASH_CE2 * L_PW * SBHE_L
+ PCMD * FLASH_CE3 * L_PW * SBHE_L
+ PCMD * FLASH_CE4 * L_PW * SBHE_L
+ PCMD * FLASH_CE5 * L_PW * SBHE_L
+ PCMD * FLASH_CE5 * L_PW * RS_CEf * RS_CE->SBHE_L 3-S %
+ PCMD * Modes_CE * L_PW * SBHE_L;
```



```

_WT_L'   = PCMD * FLASH_CE0 * L_PW * /SA00
          + PCMD * FLASH_CE1 * L_PW * /SA00
          + PCMD * FLASH_CE2 * L_PW * /SA00
          + PCMD * FLASH_CE3 * L_PW * /SA00
          + PCMD * FLASH_CE4 * L_PW * /SA00
          + PCMD * FLASH_CE5 * L_PW * /SA00
          + PCMD * Modes_CE * L_PW * /SA00;

```

```

iWT      = PCMD * FLASH_CE0 * L_PW
          + PCMD * FLASH_CE1 * L_PW
          + PCMD * FLASH_CE2 * L_PW
          + PCMD * FLASH_CE3 * L_PW
          + PCMD * FLASH_CE4 * L_PW
          + PCMD * FLASH_CE5 * L_PW
          + PCMD * Modes_CE * L_PW;

```

```

_HOLD'   = Hold;

```

```

iDL_IN    = PCMD * FLASH_CE0
           + PCMD * FLASH_CE1
           + PCMD * FLASH_CE2
           + PCMD * FLASH_CE3
           + PCMD * FLASH_CE4
           + PCMD * FLASH_CE5
           + PCMD * Modes_CE;

```

```

END$

```

```

*****

```

292097-24

5AC312 EPLD

OPTIONS: TURBO = OFF

PART: N5AC312 % PLCC %

%

This EPLD contains logic for the following:

Flash Array VPP control

FDM and XIP Power Down control

Flash Disk Memory READY Interrupt control

MD_REV_A - 04/08/92 Initial release

MD_REV_B - 04/29/92 MODES_CE now a function for FLASH_CES and

RS_CE, rather than separate. This is to
save pins on the other EPLD.

%

INPUTS:

WT#@2 % Write Strobe %
RD#@4 % Read Strobe %
SA01@9 % Buffered/Latched Address bit 01 %
SA02@10 % Buffered/Latched Address bit 02 %
RSTDRV@16 % ISA bus Reset Driver %
ALL_RDY@27 % All Flash ICs assigned to Flash Disk are RDY %
SMOUT4@12 % +5LOCAL ON when this is high %
FLASH_CE5#@6
RS_CE@8 % ROM Stub Chip Select %

OUTPUTS:

% Output pins %

EN_FLASH_VPP@22 % Pwr Cntrl Reg bit 00; Enable Flash Array VPP %
PWR_UP@26 % Pwr Cntrl Reg /bit 03 %
PWRD#@3 % Power control signal to entire Flash Array %
INTR@13 % Intr Cntrl Reg bit 00; Flash Disk RDY Intr %
SD00@17 % Internal Data Bus %
SD01@18
SD02@20
SD03@21
UNUSED23@23
UNUSED24@24

% Buried Macrocells %

CLR_INTRF % Clear Interrupt flip-flop %
EN_INTRF % Modes Reg bit 02; Enable Flash Disk RDY Intr %

NETWORK:

nWT = INP(WT#)
nRD = INP(RD#)
SA01 = INP(SA01)
SA02 = INP(SA02)
RSTDRV = INP(RSTDRV)
ALL_RDY = INP(ALL_RDY)
SMOUT4 = INP(SMOUT4)
INTR_CLK = CLKB(ALL_RDY)
nFLASH_CE5 = INP(FLASH_CE5#)
RS_CE = INP(RS_CE)

SD00,SD00f = COIF(iSD00,CHIP_RD)
SD01,SD01f = COIF(iSD01,CHIP_RD)
SD02,SD02f = COIF(iSD02,CHIP_RD)

SD03,SD03f = COIF(iSD03,CHIP_RD)

EN_FLASH_VPP,EN_FLASH_VPPf = RORF(iEN_FLASH_VPP,nWT,RSTDRV,GND,VCC)
PWR_UP,PWR_UPf = RORF(iPWR_UP,nWT,GND,RSTDRV,VCC)
PWRD# = CONF(PWRD,VCC)
EN_INTRf = NORF(iEN_INTR,nWT,RSTDRV,GND)
INTR,INTRf = RORF(VCC,INTR_CLK,CLR_INTRf,GND,VCC)
CLR_INTRf = NOCF(iCLR_INTR)
UNUSED23 = CONF(GND,VCC)
UNUSED24 = CONF(GND,VCC)

EQUATIONS:

% Internal declarations %

RD = nRD;
WT = nWT;
FLASH_CE5 = nFLASH_CE5;
Modes_CE = RS_CE * /FLASH_CE5;
Sel_Pwr_Reg = Modes_CE * /SA02 * /SA01;
Sel_Intr_Reg = Modes_CE * /SA02 * SA01;
Sel_Clr_Intr = Modes_CE * SA02 * SA01;
CHIP_RD = RD * Modes_CE;
FDM_Busy = /ALL_RDY;

% Outputs and buried registers %

iSD00 = Sel_Pwr_Reg * EN_FLASH_VPPf
+ Sel_Intr_Reg * EN_INTRf;
iSD01 = Sel_Pwr_Reg * GND
+ Sel_Intr_Reg * INTRf;
iSD02 = Sel_Pwr_Reg * GND
+ Sel_Intr_Reg * FDM_Busy;
iSD03 = Sel_Pwr_Reg * /PWR_UPf
+ Sel_Intr_Reg * GND;
iEN_FLASH_VPP = Sel_Pwr_Reg * SD00f
+ /Sel_Pwr_Reg * EN_FLASH_VPPf;
iPWR_UP = Sel_Pwr_Reg * /SD03f


```

+ /Sel_Pwr_Reg * PWR_UPf;

_PWRD      = PWR_UPf
+ SMOUT4;

iEN_INTR   = Sel_Intr_Reg * SD00f
+ /Sel_Intr_Reg * EN_INTRf;

iCLR_INTR  = RSTDRV
+ /EN_INTRf
+ WT * Sel_Clr_Intr;

END$

```

292097-35

+ 1241 Pwr Reg * PWR_UP1

PWRD - PWR_UP1

+ 1241 Pwr Reg * PWR_UP1

EN_DTR - 241 Pwr Reg * 24001

+ 1241 Pwr Reg * EN_DTR

CLR_DTR - RSTDRV

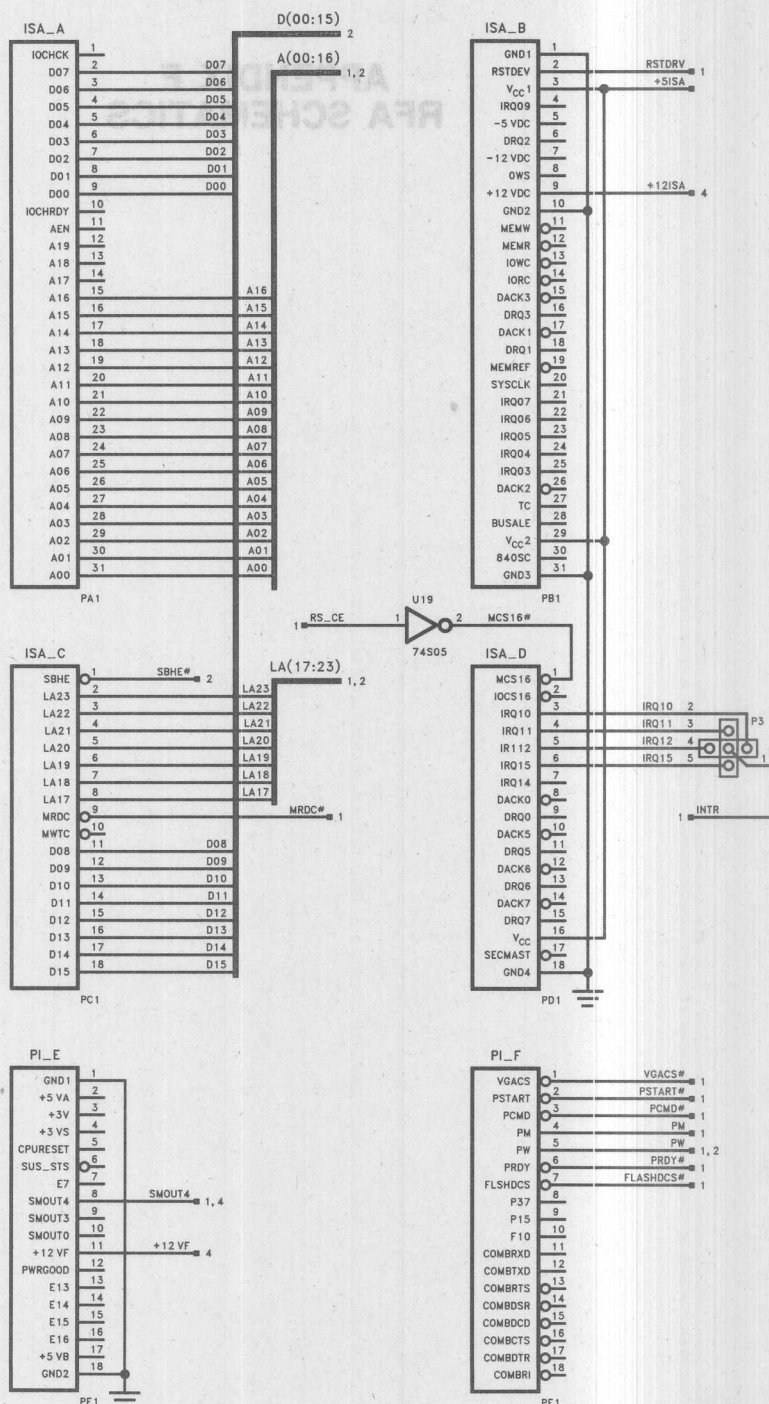
+ 1241 Pwr Reg * EN_DTR

+ 1241 Pwr Reg * 241 Pwr Reg

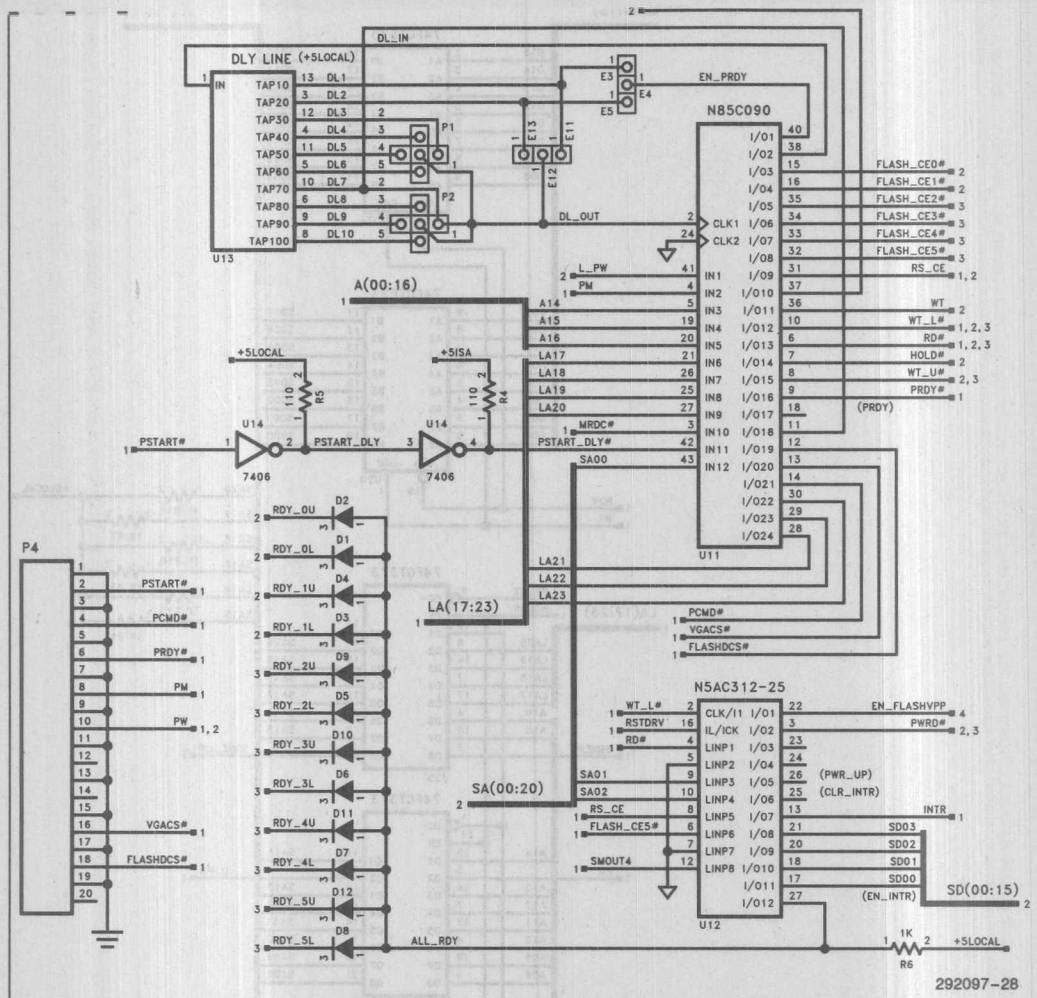
END

END

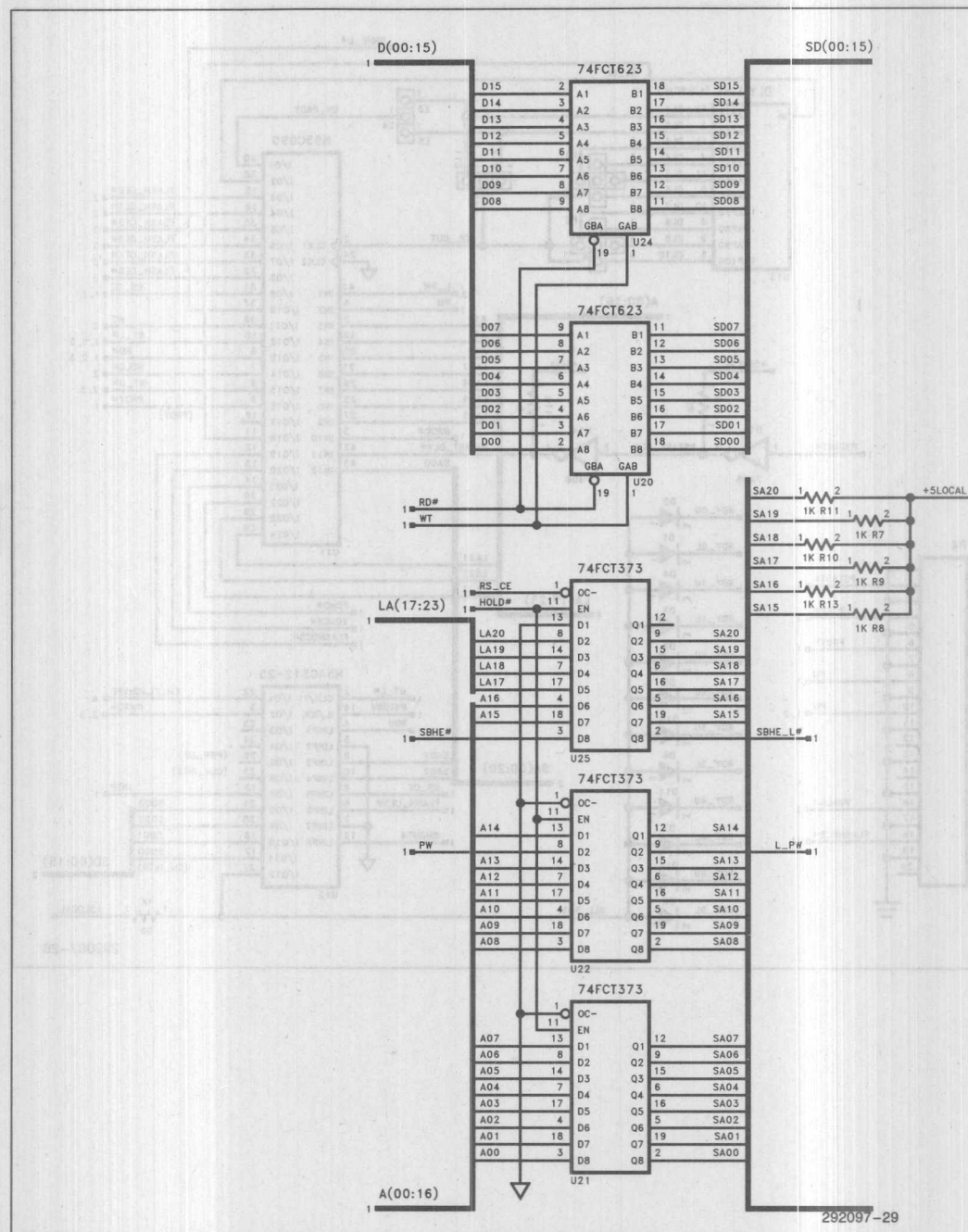
3



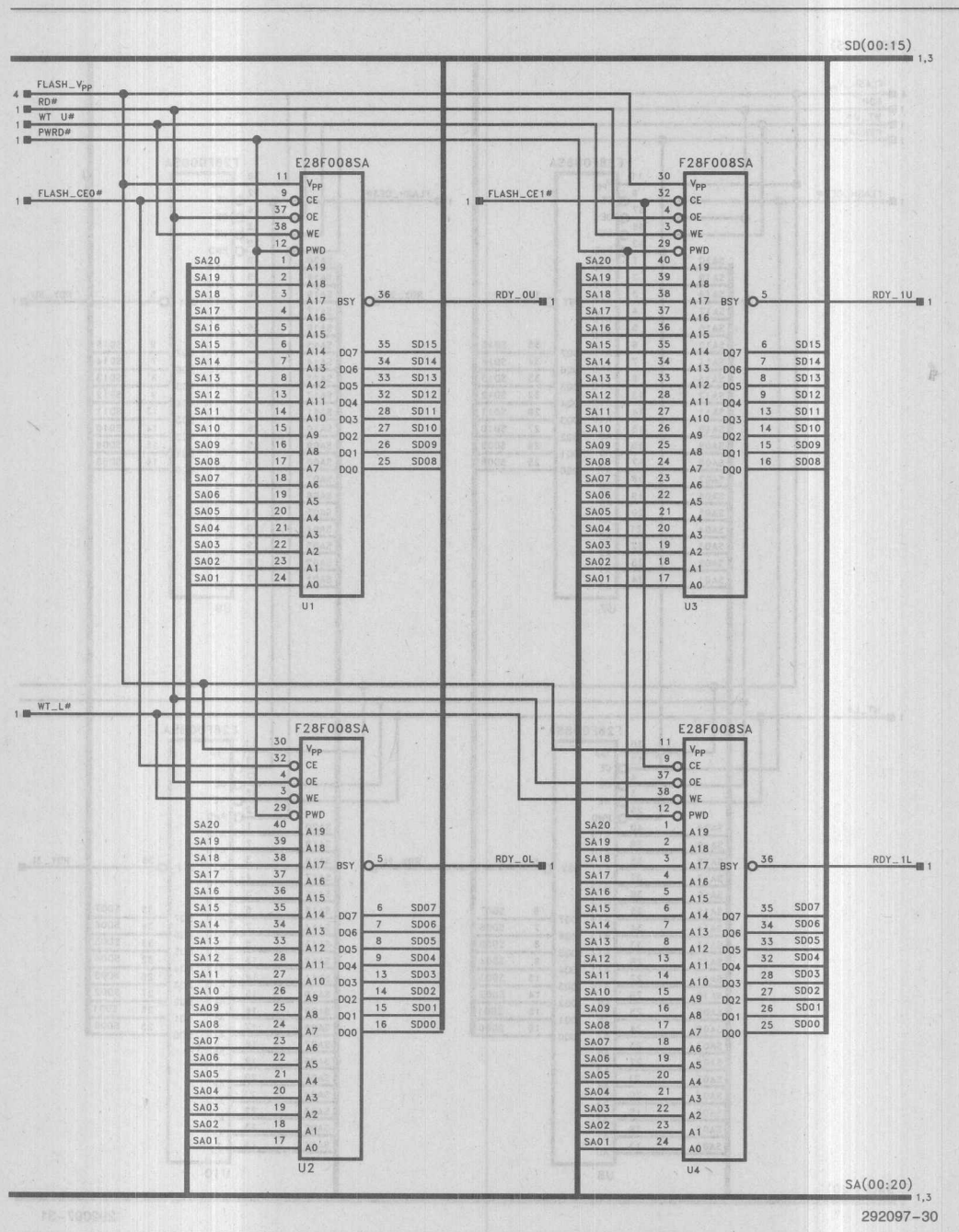
292097-27



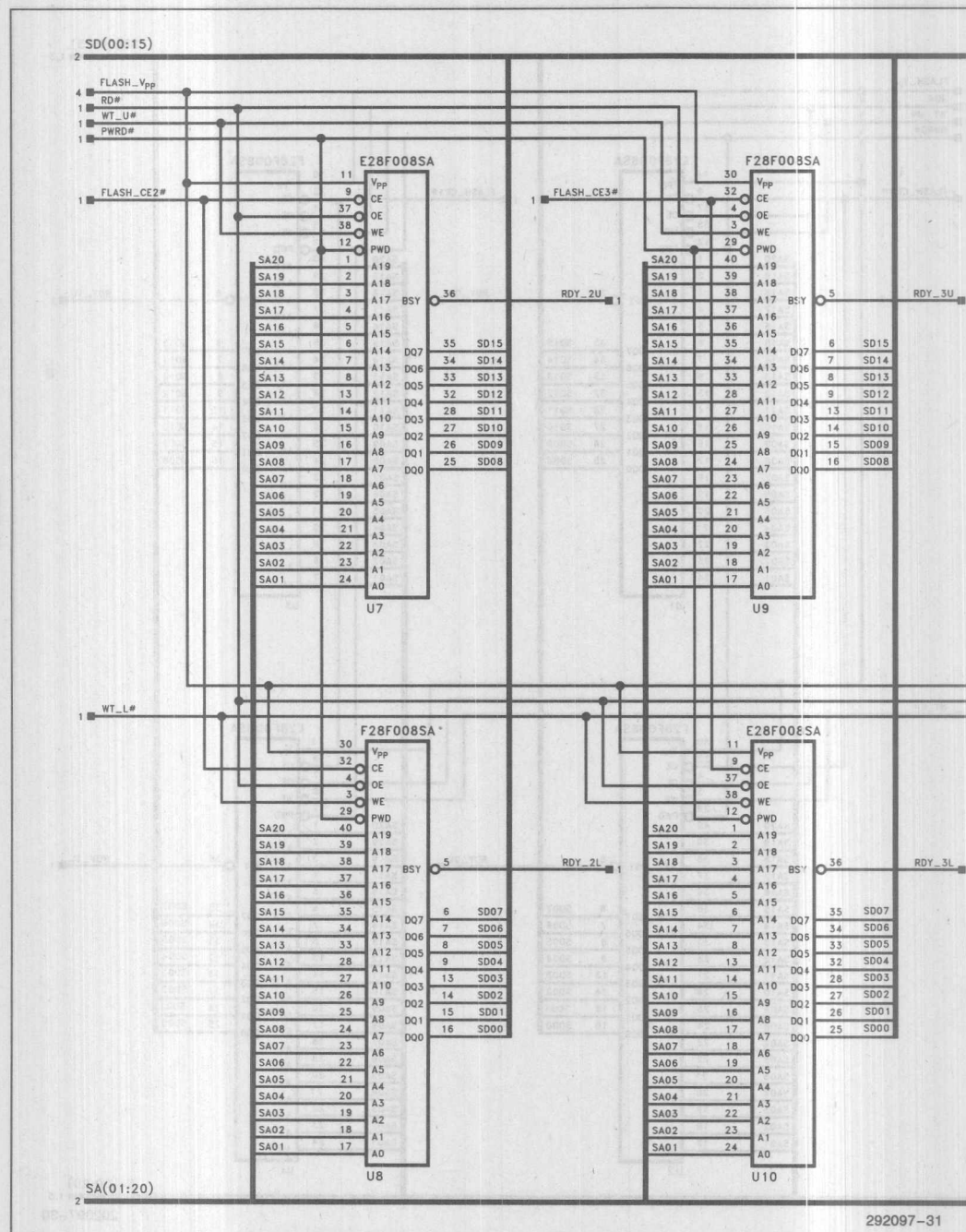
3

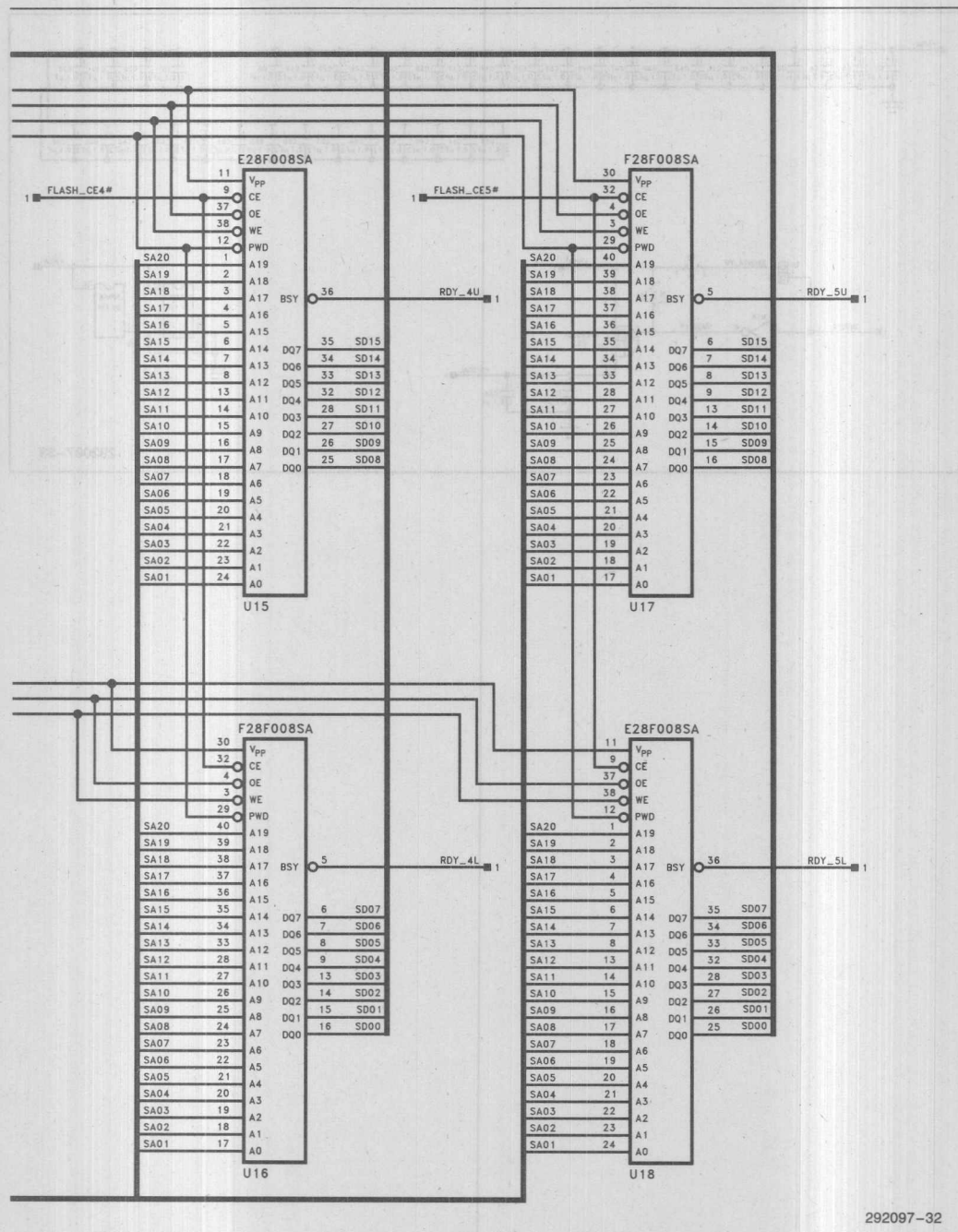


292097-29

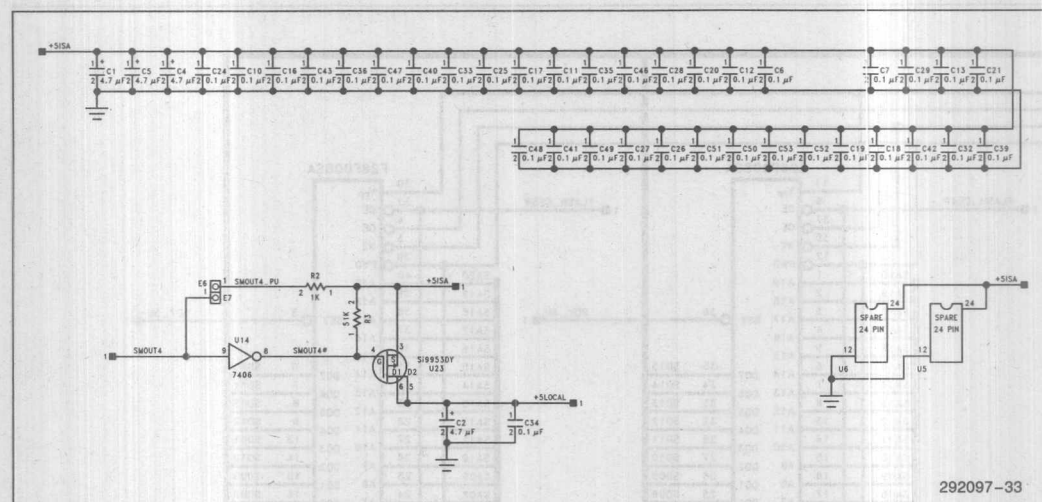


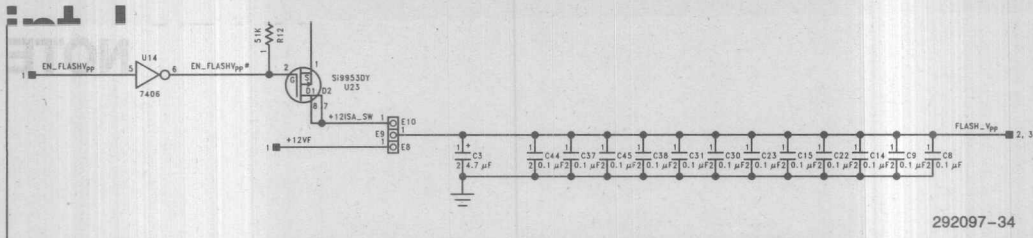
3





3





292097-34

APPLICATION
NOTE

28F008SA
Automation and Algorithms

BRIAN DIPERT
MCD MARKETING APPLICATIONS

September 1993

28F008SA Automation and Algorithms

CONTENTS	PAGE
1.0 INTRODUCTION	3-196
2.0 AUTOMATION AND ALGORITHMS	3-196
2.1 28F008SA Automation and the Write State Machine	3-196
Command User Interface	
Status Register	
2.2 Byte Write Algorithm	3-199

Figure 1 shows a block diagram of the 28F008SA and its internal contents. Although a main subject of this application note is software instructions to read and write memory, it is useful to begin with an overview of the 28F008SA hardware architecture and the internal organization of the device. In particular, the application note will first discuss the Write State Machine (WSM) and Command User Interface/Status Register, and then explain the software routines that control the hardware.

2.1 28F008SA Automation and the Write State Machine

When the system microprocessor reads flash memory data from the 28F008SA, it uses control lines CE# and OE#, along with address inputs, to select a byte of data directly from the memory cell array. However, the system does not directly access the array when it writes to the 28F008SA; instead it writes to the Command User Interface, whose register contents are interpreted and translated into WSM actions. The WSM can be thought of as a dedicated "processor" along with command clock generation circuitry, integrated into the flash memory. After receiving proper commands or command sequences, it controls the write and block erase algorithms internally. The nature of the WSM is not invisible to the system; the WSM interface to the outside world through a full-pattern status register and dedicated RYBY# (Read/Byte/Write) control. An application note has significant details, some of which are more obvious than others.

CONTENTS	PAGE
2.3 Block Erase Algorithm	3-201
2.4 Erase Suspend/Resume Algorithm	3-201
2.5 Write State Machine Current/Next State Overview	3-204
Read Array	
Byte Write Setup	
Byte Write	
Erase Setup	
Erase	
Erase Command Error	
Erase Suspend to Status/Array	
Read Status	
Read Identifier	
2.6 Block Erase As a Background Task	3-206

ADDITIONAL INFORMATION	3-207
------------------------------	-------

- Hardware Data Protection Features
- Erase/Write Lockout during Power Transitions
- Industry Standard Packaging
- 40 and 130V 44 Lead PSOP
- ETOX III: Nonvolatile Flash Memory Technology
- 12V Byte Write/Block Erase

The 28F008SA's automation is a significant enhancement to the various algorithms of first-generation flash memory devices. System software and hardware designers that fully understand and exploit this automation will greatly benefit from its versatility and capabilities. The concepts presented in this document are applicable to each design.

This application note discusses in-depth operation of the 28F008SA's Flash Write State Machine and internal algorithms, emphasizing how they interface to system hardware and software. The 28F008SA datasheet (document number 280129) is a valuable reference.

1.0 INTRODUCTION

The Intel 28F008SA FlashFile™ Memory is today's optimum solution for high density solid state storage. Flash memory, exemplified by the 28F008SA, is an enabling technology for today's powerful system designs that are higher performance, more compact, lighter, more rugged and have longer battery life.

Features of the 28F008SA include:

- High-Density Symmetrically Blocked Architecture:
 - Sixteen 64-Kbyte Blocks
- Extended Cycling Capability
 - 100,000 Block Erase Cycles
 - 1.6 Million Block Erase Cycles per Chip
- Automated Byte Write and Block Erase
 - Command User Interface
 - Status Register
- System Performance Enhancements
 - RY/BY# Status Output
 - Erase Suspend Capability
- Deep Powerdown Mode
 - 0.20 μ A I_{CC} Typical
- Very High Performance Read
 - 85 ns Maximum Access Time
- SRAM-Compatible Write Interface
- Hardware Data Protection Features
 - Erase/Write Lockout during Power Transitions
- Industry Standard Packaging
 - 40 Lead TSOP, 44 Lead PSOP
- ETOX III Nonvolatile Flash Memory Technology
 - 12V Byte Write/Block Erase

The 28F008SA's automation is a significant enhancement to the manual algorithms of first-generation flash memory devices. System software and hardware designs that fully understand and exploit this automation will greatly benefit from its versatility and capabilities. The concepts presented in this document are applicable to such designs.

This application note discusses in-depth operation of the 28F008SA FlashFile memory Write State Machine and internal algorithms, emphasizing how they interface to system hardware and software. The 28F008SA datasheet (order number 290429) is a valuable reference

document, providing in-depth device technical specifications, package pinouts and timing waveforms. Companion application note AP-359, "28F008SA Hardware Interfacing" (order number 292094) describes supply voltage derivation and filtering, control input/output implementation, high density layout and high speed design techniques, as well as providing example system interfaces to common microprocessor buses. AP-360, "28F008SA Software Drivers" (order number 292095) provides example ASM-86 and "C" routines for controlling the 28F008SA. AP-359 and AP-360 should be reviewed in conjunction with this application note and the 28F008SA datasheet for a complete understanding of this device.

2.0 AUTOMATION AND ALGORITHMS

Figure 1 shows a block diagram of the 28F008SA and its internal contents. Although a main subject of this application note is software interface to read and alter memory contents, it is useful to begin with an overview of the 28F008SA hardware subsections that are directly manipulated by the system. In particular, this application note will first discuss the Write State Machine (WSM) and Command User Interface/Status Register, and then explain the software routines that control this hardware.

2.1 28F008SA Automation and the Write State Machine

When the system microprocessor reads flash memory data from the 28F008SA, it uses control lines CE# and OE#, along with address inputs, to select a byte of data directly from the memory cell array. However, the system does not directly access the array when it writes to the 28F008SA; instead it writes to the Command User Interface, whose register contents are interpreted and translated into WSM actions. The WSM can be thought of as a dedicated "processor", along with companion clock-generation circuitry, integrated into the flash memory. After receiving proper commands or command sequences, it controls byte write and block erase algorithms internally. The status of the WSM is not invisible to the system; the WSM interfaces to the outside world through a full-featured Status Register and dedicated RY/BY# (Ready/Busy#) output. Automation has significant benefits, some of which are more obvious than others.

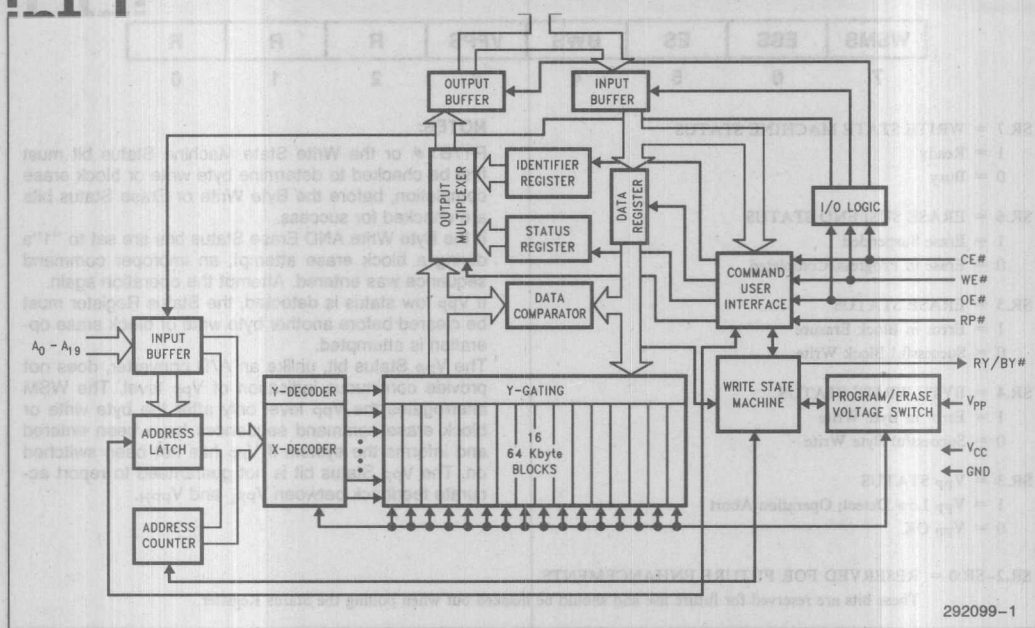


Figure 1. 28F008SA Block Diagram

The WSM architecture dramatically simplifies the program and erase algorithms of first-generation flash memory devices. Hardware/software timers, erase pre-programming, byte-by-byte verification and margining, pulse repetition and limited microprocessor multitasking capability throughout data update have been eliminated, replaced by a simple two-command write for both block erase and byte write. The 28F008SA WSM halts itself when its internal algorithms are complete, and can alert the system to this completion by a hardware interrupt (using RY/BY#) or via software polling of the 28F008SA Status Register.

Internal automation frees the system to execute higher-priority tasks while a 28F008SA is being block erased or byte written, and inherent in this capability is the most powerful advantage of the WSM. Operating systems prioritize file operations in the following order:

- Read
- Write
- Erase

When an array of 28F008SA components is used as solid-state storage (in a memory card, integrated in a flash-based "hard drive" form factor or resident on the system motherboard), system software can initiate slower block erase (0.3 sec minimum) of one or several

28F008SAs and, by not being "tied" to the erase algorithm, execute higher priority reads (85 ns minimum) or writes (6 μ s minimum) of other 28F008SAs as operating system requests dictate. Additionally, erase suspend/resume capability allows data retrieval from a 28F008SA currently being block erased, again enabling "read" as the highest priority task. Block erase as a background task is discussed in Section 2.6 of this document.

Command User Interface

Table 2 shows the various command sequences that are accepted and interpreted by the 28F008SA Command User Interface and WSM. Writes to the CUI enable reading of device data and intelligent identifiers, reading and clearing of the Status Register, and commencement of internal byte write, block erase and erase suspend/resume algorithms. The CUI itself does not occupy a specifically addressable memory location, and contains a latch used to store the command and address/data information needed to execute the command. Erase Setup and Erase Confirm commands require both appropriate command data and an address within the block to be erased. The Byte Write Setup command requires appropriate command data and the address of the location to be written, while the Byte Write command consists of the data to be written and its address location.

Table 1. Status Register Definitions

WSMS	ESS	ES	BWS	VPPS	R	R	R
7	6	5	4	3	2	1	0

SR.7 = WRITE STATE MACHINE STATUS
 1 = Ready
 0 = Busy

SR.6 = ERASE SUSPEND STATUS
 1 = Erase Suspended
 0 = Erase in Progress/Completed

SR.5 = ERASE STATUS
 1 = Error in Block Erasure
 0 = Successful Block Write

SR.4 = BYTE WRITE STATUS
 1 = Error in Byte Write
 0 = Successful Byte Write

SR.3 = V_{PP} STATUS
 1 = V_{PP} Low Detect; Operation Abort
 0 = V_{PP} OK

SR.2-SR.0 = RESERVED FOR FUTURE ENHANCEMENTS
 These bits are reserved for future use and should be masked out when polling the Status Register.

NOTES:
 RY/BY# or the Write State Machine Status bit must first be checked to determine byte write or block erase completion, before the Byte Write or Erase Status bits are checked for success.
 If the Byte Write AND Erase Status bits are set to "1"s during a block erase attempt, an improper command sequence was entered. Attempt the operation again.
 If V_{PP} low status is detected, the Status Register must be cleared before another byte write or block erase operation is attempted.
 The V_{PP} Status bit, unlike an A/D converter, does not provide continuous indication of V_{PP} level. The WSM interrogates the V_{PP} level only after the byte write or block erase command sequences have been entered and informs the system if V_{PP} has not been switched on. The V_{PP} Status bit is not guaranteed to report accurate feedback between V_{PP}_L and V_{PP}_H.

Table 2. Command Definitions

Command	Bus Cycles Req'd	Notes	First Bus Cycle			Second Bus Cycle		
			Operation	Address	Data	Operation	Address	Data
Read Array/Reset	1		Write	X	FFH			
Intelligent Identifier	3	1, 2, 3	Write	X	90H	Read	IA	IID
Read Status Register	2	2	Write	X	70H	Read	X	SRD
Clear Status Register	1		Write	X	50H			
Erase Setup/Erase Confirm	2	1	Write	BA	20H	Write	BA	D0H
Erase Suspend/Erase Resume	2		Write	X	B0H	Write	X	D0H
Byte Write Setup/Write	2	1, 2, 4	Write	WA	40H	Write	WA	WD
Alternate Byte Write Setup/Write	2	1, 2, 4	Write	WA	10H	Write	WA	WD

NOTES:

1. IA = Identifier Address: 00H for manufacturer code, 01H for device code.
 BA = Address within the block being erased.
 WA = Address of memory location to be written.
2. SRD = Data read from Status Register. See Table 4 for a description of the Status Register bits.
 WD = Data to be written at location WA. Data is latched on the rising edge of WE.
 IID = Data read from intelligent identifiers.
3. Following the intelligent identifier command, two read operations access manufacturer and device codes.
4. Either 40H or 10H are recognized by the WSM as the Byte Write Setup command.
5. Commands other than those shown above are reserved by Intel for future device implementations and should not be used.

Status Register

Table 1 shows the 28F008SA Status Register and defines its various bits. Like the Command User Interface, it does not occupy a specific memory location within the device. It functions as an output of the WSM, informing the system when internal byte write or block erase algorithms have completed, if these algorithms completed successfully, and whether the 28F008SA is currently in Erase Suspend mode. Bit 7 (Write State Machine Status) is replicated in the device RY/BY# hardware output. The default state of the upper 5 bits of the Status Register after powerup and return from deep powerdown mode is 10000 (binary).

A separate Clear Status Register command allows re-initialization of Status Register data after analysis. The Status Register is not cleared until this command is written to the 28F008SA.

Bits 5 and 4 of the Status Register, if set by the WSM via a byte write or block erase attempt, do not block subsequent attempts (they need not be cleared before another byte write/block erase command sequence is written to the device). However, if the WSM detects a "low Vpp" condition and subsequently sets bit 3 of the Status Register, the Status Register MUST be cleared before another algorithm command sequence will be recognized by the 28F008SA.

It is important to note that the Vpp Status bit of the Status Register DOES NOT act like an always-functional A/D converter; its normal state, even with Vpp below 6.5V, is "0". The WSM only analyzes the Vpp level after a byte write or block erase command sequence has been written to the device, and if it detects that Vpp is "low" it will cancel the impending byte write or block erase operation and set the Vpp Status bit to "1". Therefore, the Vpp Status bit cannot be used by the system as an indication of proper Vpp level, before a byte write or block erase sequence is initiated. The system should instead insert an appropriate software delay between turning on Vpp and writing an initial command sequence, or use external hardware as a Vpp feedback mechanism.

2.2 Byte Write Algorithm

Figure 2 provides a graphical representation of the 28F008SA byte write algorithm. As can be seen, this consists solely of a two-command write sequence, followed by a periodic poll of the device RY/BY# output or Status Register. The 28F008SA automatically outputs Status Register data when read after the two-command byte write sequence (see Section 2.5). Byte write typically completes in 9 μ s.

The byte write algorithm requires high voltage VppH ($12V \pm 5\%$) on the device Vpp input until internal algorithm completion is reported by the WSM. If byte write is attempted while $V_{pp} = V_{ppl} (\leq 6.5V)$, the Vpp Status bit of the Status Register will be set to "1", and array data will not be altered. Byte write attempts while $V_{ppl} < V_{pp} < V_{ppH}$ produce spurious results and should not be attempted.

The Status Register will only report errors for "1"s that do not write to "0"s during a byte write attempt. Erasure (see Section 2.3) is the method used to change data "0"s to "1"s using flash technology. If the system software attempts to write "1"s to a byte at bit locations already at value "0", no Status Register error will be reported for those specific bits.

It is often desired to write multiple bytes of data at one time to memory. Since the Status Register is only cleared after the Clear Status Register command is written to the 28F008SA, a string of bytes can be sequentially written to the device before the "full status check procedure" examines Status Register bits other than SR.7.

Byte write abort occurs when the 28F008SA RP# (Reset/Powerdown) input drops to VIL (deep powerdown mode is entered), or Vpp drops to Vppl. Although the WSM is halted in either case, byte data is partially written at the location where aborted. A repeat byte write sequence after system integrity is restored will complete the desired operation, or data can be initialized to a known value of "FF" thru block erasure.

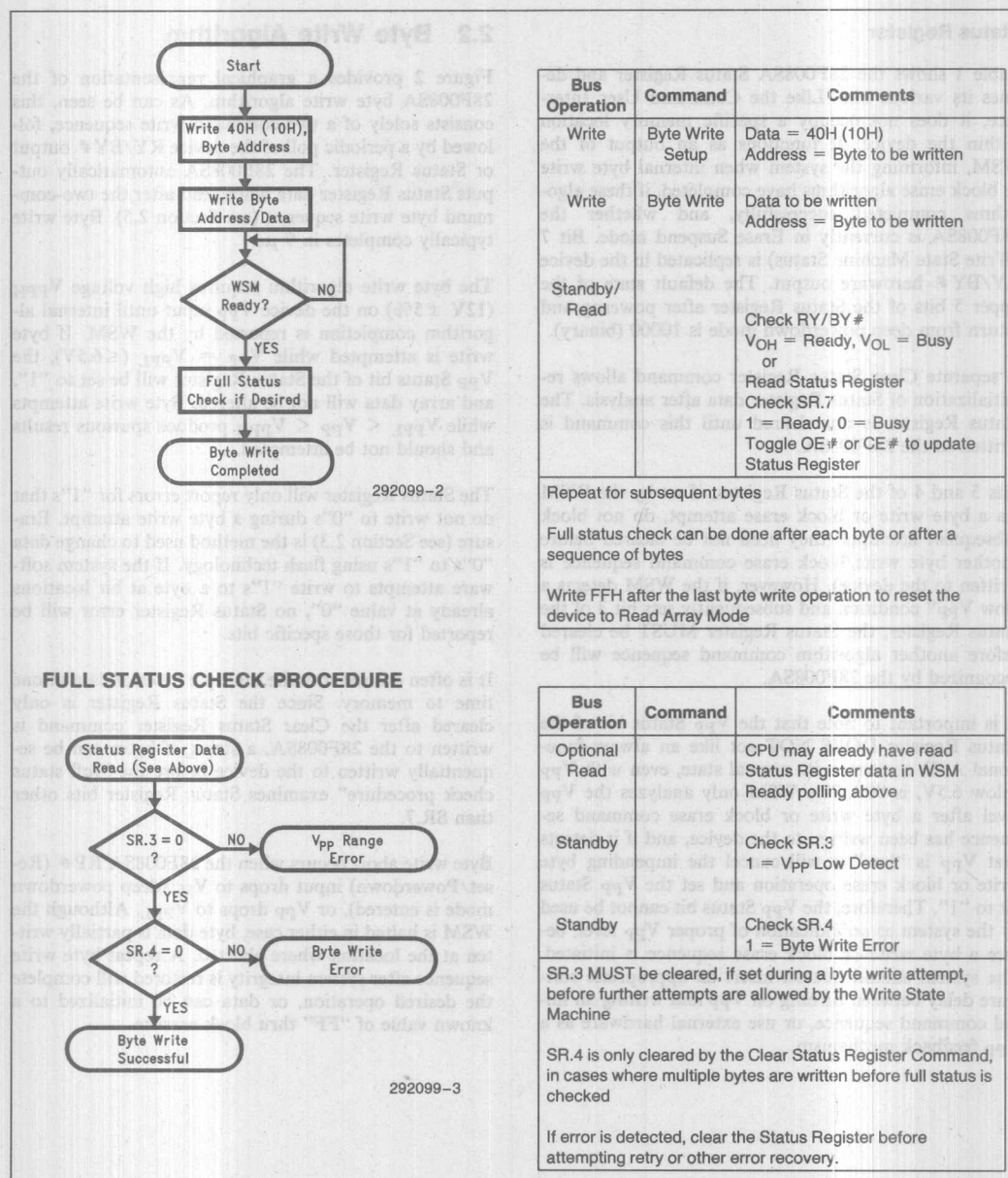


Figure 2. Automated Byte Write Flowchart

Figure 3 provides a graphical representation of the 28F008SA block erase algorithm, similar in its two-command write sequence to the byte write algorithm discussed earlier. Both the Erase Setup and Erase Confirm commands must be accompanied by an address within the desired block to be erased to FFH. The 28F008SA automatically outputs Status Register data when read after the two-command block erase sequence (see Section 2.5). Block erase typically completes in 1.6 sec.

Again similar to byte write, the block erase algorithm requires high voltage V_{PPH} ($12V \pm 5\%$) on the device V_{PP} input until internal algorithm completion is reported by the WSM. If block erase is attempted while $V_{PP} = V_{PPL}$ ($\leq 6.5V$), the V_{PP} Status bit of the Status Register will be set to "1", and array data will not be altered. Block erase attempts while $V_{PPL} < V_{PP} < V_{PPH}$ produce spurious results and should not be attempted.

If write of the Erase Setup command is followed by write of any other command but Erase Confirm, the WSM will decode this as an illegal sequence. It will not attempt to erase the specified block, and will report error back to the system by setting both the Erase Status and Byte Write Status bits of the Status Register to "1". Since the Status Register is only cleared after the Clear Status Register command is written to the 28F008SA, a string of blocks within a 28F008SA can be sequentially erased before the "full status check procedure" examines Status Register bits other than SR.7.

down mode is entered) or V_{PP} drops to V_{PPL} . After a block erase sequence after system integrity is restored will complete the desired operation.

2.4 Erase Suspend/Resume Algorithm

Figure 4 gives a software flowchart for implementing erase suspend/resume using the 28F008SA. As mentioned in Section 2.1, operating systems prioritize data reads highest, and consequently the 28F008SA has been designed with read as its highest performance function. Erase suspend allows system software to postpone WSM-controlled block erase if the system requests read of data from a **different** block of the same device. Although any block of the 28F008SA can be read, the block being erased when suspended will contain unknown data.

The 28F008SA is suspended by writing the Erase Suspend command (BOH) to it while the WSM is executing an erase algorithm. The WSM will halt block erase, set bits 7 and 6 of the Status Register to "1" and transition RY/BY# to V_{OH} , after which time system software can read data from either the array or Status Register. Issuing the Erase Resume command (DOH) signals the WSM to resume block erase.

V_{PP} must remain at V_{PPH} throughout the erase suspend interval, even when reading from the flash memory array. The 28F008SA will detect a V_{PP} transition to V_{PPL} while suspended, and report this error via Status Register bit 3 (set to "1") after the Erase Resume command is written to it.

3

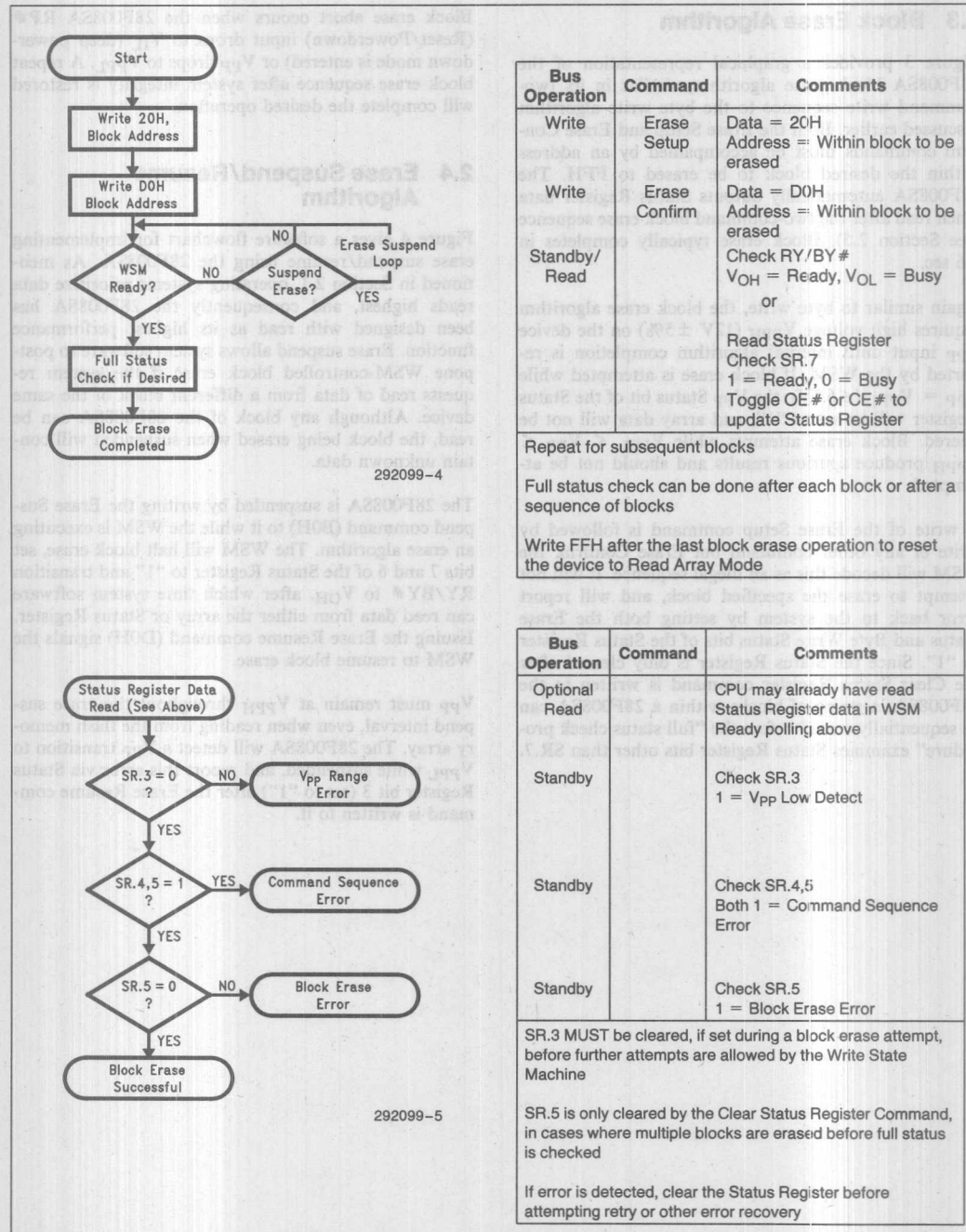


Figure 3. Automated Block Erase Flowchart

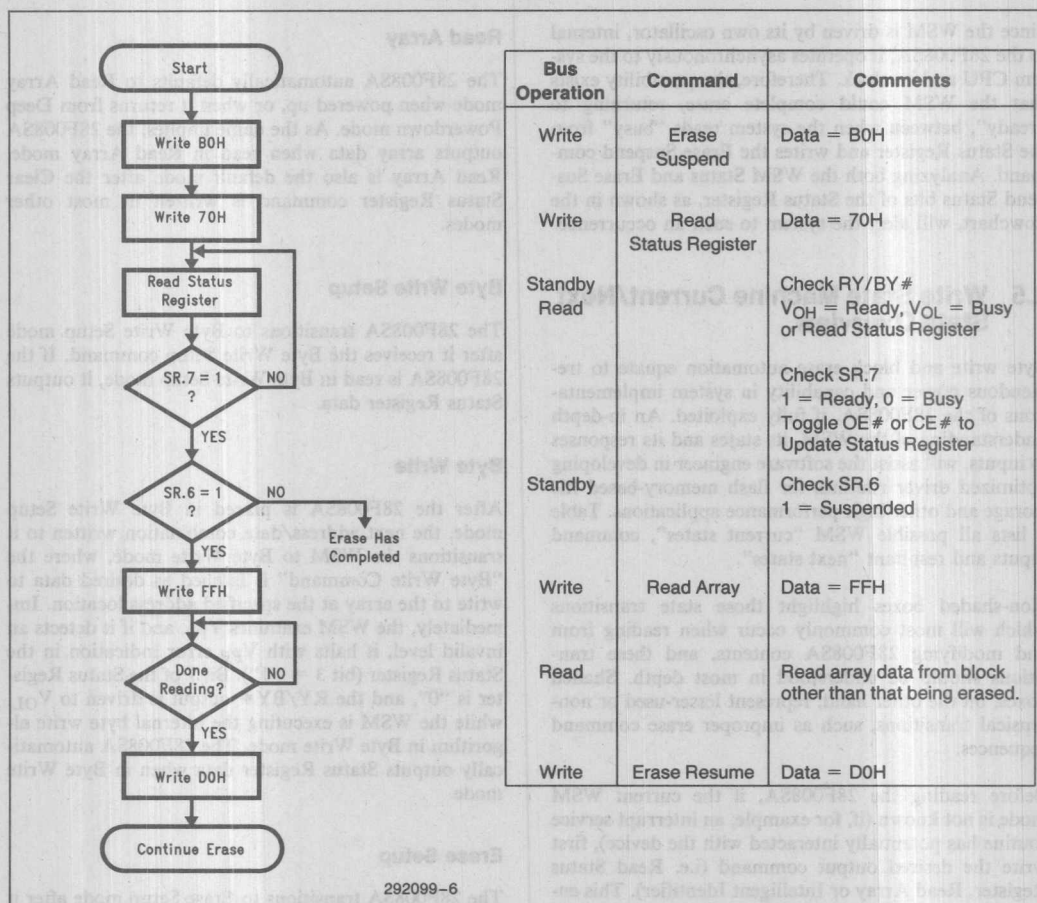


Figure 4. Erase Suspend/Resume Flowchart

Since the WSM is driven by its own oscillator, internal to the 28F008SA, it operates asynchronously to the system CPU and its clock. Therefore, the possibility exists that the WSM could complete erase, returning to "ready", **between** when the system reads "busy" from the Status Register and writes the Erase Suspend command. Analyzing both the WSM Status and Erase Suspend Status bits of the Status Register, as shown in the flowchart, will alert the system to such an occurrence.

2.5 Write State Machine Current/Next State Overview

Byte write and block erase automation equate to tremendous power and capability in system implementations of the 28F008SA, if fully exploited. An in-depth understanding of the WSM, its states and its responses to inputs, will assist the software engineer in developing optimized driver routines for flash memory-based file storage and other high-performance applications. Table 3 lists all possible WSM "current states", command inputs and resultant "next states".

Non-shaded boxes highlight those state transitions which will most commonly occur when reading from and modifying 28F008SA contents, and these transitions should be understood in most depth. Shaded boxes, on the other hand, represent lesser-used or non-sensical transitions, such as improper erase command sequences.

Before reading the 28F008SA, if the current WSM mode is not known (if, for example, an interrupt service routine has potentially interacted with the device), first write the desired output command (i.e. Read Status Register, Read Array or Intelligent Identifier). This ensures that the 28F008SA will be in a known state when read and will output expected data.

Read Array

The 28F008SA automatically defaults to Read Array mode when powered up, or when it returns from Deep Powerdown mode. As the name implies, the 28F008SA outputs array data when read in Read Array mode. Read Array is also the default mode after the Clear Status Register command is written in most other modes.

Byte Write Setup

The 28F008SA transitions to Byte Write Setup mode after it receives the Byte Write Setup command. If the 28F008SA is read in Byte Write Setup mode, it outputs Status Register data.

Byte Write

After the 28F008SA is placed in Byte Write Setup mode, the next address/data combination written to it transitions the WSM to Byte Write mode, where the "Byte Write Command" is latched as desired data to write to the array at the specified address location. Immediately, the WSM examines V_{pp} , and if it detects an invalid level, it halts with V_{pp} error indication in the Status Register (bit 3 = "1"). Bit 7 of the Status Register is "0", and the RY/BY# output is driven to V_{OL} , while the WSM is executing the internal byte write algorithm in Byte Write mode. The 28F008SA automatically outputs Status Register data when in Byte Write mode.

Erase Setup

The 28F008SA transitions to Erase Setup mode after it receives the Erase Setup command. If the 28F008SA is read in Erase Setup mode, it outputs Status Register data.

Table 3. Write State Machine Current/Next States

Current State	RY/BY # Status	Data When Read	Command Input (and Next State)								
			Read Array (FFH)	Byte Write Setup (10/40H)	Erase Setup (20H)	Erase Confirm (D0H)	Erase Suspend (B0H)	Erase Resume (D0H)	Read Status (70H)	Clear Status (50H)	Read ID (90H)
Read Array	V _{OH}	Array	Read Array	Byte Write Setup	Erase Setup	Read Array	Read Array	Read Array	Read Status	Read Array	Read ID
Byte Write Setup	V _{OH}	Status	Byte Write (Command Input = Data to be Byte Written)								
Byte Write (Not Complete)	V _{OL}	Status	Byte Write								
Byte Write (Complete)	V _{OH}	Status	Read Array	Byte Write Setup	Erase Setup	Read Array			Read Status	Read Array	Read ID
Erase Setup	V _{OH}	Status	Erase Command Error			Erase	Erase Command Error	Erase	Erase Command Error		
Erase Command Error	V _{OH}	Status	Read Array	Byte Write Setup	Erase Setup	Read Array			Read Status	Read Array	Read ID
Erase (Not Complete)	V _{OL}	Status	Erase				Erase Suspend to Status	Erase			
Erase (Complete)	V _{OH}	Status	Read Array	Byte Write Setup	Erase Setup	Read Array			Read Status	Read Array	Read ID
Erase Suspend to Status	V _{OH}	Status	Erase Suspend to Array	Reserved	Erase Suspend to Array	Erase	Erase Suspend to Array	Erase	Erase Suspend to Status	Erase Suspend to Array	Reserved
Erase Suspend to Array	V _{OH}	Array	Erase Suspend to Array	Reserved	Erase Suspend to Array	Erase	Erase Suspend to Array	Erase	Erase Suspend to Status	Erase Suspend to Array	Reserved
Read Status	V _{OH}	Status	Read Array	Byte Write Setup	Erase Setup	Read Array			Read Status	Read Array	Read ID
Read Identifier	V _{OH}	ID	Read Array	Byte Write Setup	Erase Setup	Read Array			Read Status	Read Array	Read ID

NOTE:

1. State transitions labeled "Reserved" are set aside by Intel Corporation for potential future device implementations. Command sequences to access these states should not be attempted.

Erase

After the 28F008SA is placed in Erase Setup mode, write of the Erase Confirm command transitions the WSM to Erase mode, where the specified address is decoded into one of 16 blocks to be erased. Immediately, the WSM examines V_{pp} , and if it detects an invalid level, it halts with V_{pp} error indication in the Status Register (bit 3 = "1"). Bit 7 of the Status Register is "0", and the RY/BY# output is driven to V_{OL} , while the WSM is executing the internal block erase algorithm in Erase mode. The 28F008SA automatically outputs Status Register data when in Erase mode.

Erase Command Error

This is the other possible transition mode after Erase Setup, and occurs when an invalid command (anything but Erase Confirm/Resume) is written to the 28F008SA as the second in the two-command block erase sequence. In this mode, the WSM does not attempt a block erase, and it returns an error indication to the system by setting both bits 4 and 5 of the Status Register to "1". The 28F008SA automatically outputs Status Register data when in Erase Command Error mode.

Erase Suspend to Status/Array

While the WSM is busy executing an internal block erase algorithm, it can be placed in erase suspend by writing the Erase Suspend command. After receiving and decoding this command, the WSM suspends block erase, drives the RY/BY# output to V_{OH} , sets bits 6 and 7 of the Status Register to "1" and transitions to "Erase Suspend to Status" mode. The 28F008SA automatically outputs Status Register data when in "Erase Suspend to Status" mode.

The only valid command other than Read Status and Erase Resume at this time is Read Array, which transitions the WSM to "Erase Suspend to Array" mode. As the name implies, the 28F008SA outputs array data, not Status Register contents, in this mode. While in both Erase Suspend modes, V_{pp} must remain at V_{PPH} for erase to complete successfully when resumed.

Writing the Erase Resume (same as Erase Confirm) command to the 28F008SA transitions the WSM out of Erase Suspend and back to Erase. In conjunction with this, the WSM returns RY/BY# to V_{OL} and resets bits 6 and 7 of the WSM to "0".

Read Status

As the name implies, the 28F008SA automatically outputs Status Register contents when read in Read Status mode. If system software writes the Clear Status command at this point, the WSM resets the Status Register to its default value and transitions to Read Array mode.

Read Identifier

The 28F008SA outputs its manufacturer identifier of 89H when read from address 00000H when in Read Identifier mode. Similarly, a read from address 00001H returns the device identifier A2H. Using this information, the system can automatically match the device with its proper block erase and byte write algorithms. Reads from addresses other than 00000H and 00001H are not supported by Intel, and consistent results of such reads are not documented, guaranteed or recommended.

2.6 Block Erase as a Background Task

As mentioned earlier, the internal WSM block erase algorithm typically takes 1.6 seconds to complete. Proper implementation of block erase from a hardware and software standpoint, however, can mask this delay, by taking advantage of the 28F008SA's internal automation and full-featured system interface. Execution of block erase as a background task, with higher priority read and write functions in the foreground, is the key.

The recommended scenario includes an "intelligent" operating system routine which can keep track of "busy" devices in the 28F008SA array. After initiating block erase on these components, the operating system is free to concentrate on reads and writes, or any other pending requests that demand its attention. The 28F008SA RY/BY# output alerts the system when block erase completes, and the operating system acts on this completion in the resulting interrupt service routine.

Hardware interrupt via the RY/BY# output is a recommended technique for block erase. However, this method should be evaluated closely for alerting the system to byte write completion. The WSM typically completes a byte write attempt in 9 μ s, a much shorter time than that consumed in many CPU interrupt latencies. In such cases, software polling of the 28F008SA Status Register to detect WSM "ready", versus hardware interrupt, will result in highest byte write performance. Reference AP-359, "28F008SA Hardware Interfacing", for circuit implementations that not only combine RY/BY#s into a common INT, but also allow RY/BY# masking if desired.

ADDITIONAL INFORMATION

		Order Number
28F008SA Datasheet		290429
28F008SA-L Datasheet		290435
AP-359	"28F008SA Hardware Interfacing"	292094
AP-360	"28F008SA Software Drivers"	292095
ER-27	"The Intel 28F008SA Flash Memory"	294011
ER-28	"ETOX-III Flash Memory Technology"	294012

REVISION HISTORY

Number	Description
003	PWD # pin renamed RP # to match JEDEC standards.



AP-375

APPLICATION NOTE

Upgrade Considerations from the 28F008SA to the 28F016SA

SALIM FEDEL
FLASH MEMORY APPLICATIONS ENGINEERING

October 1993

Upgrade Considerations from the 28F008SA to the 28F016SA

CONTENTS	PAGE
1.0 PURPOSE	3-210
2.0 SIMILARITIES AND DIFFERENCES BETWEEN THE 28F016SA AND THE 28F008SA	3-210
2.1 Pinout Differences	3-210
2.1.1 Hardware Compatible Configuration	3-212
2.2 Compatible Command-Set	3-213
2.3 Technology Comparison	3-214
2.4 Available Speeds	3-214
2.5 Available Packages	3-214
2.6 Operating Modes	3-214
2.7 AC Compatibility	3-214

CONTENTS	PAGE
2.8 DC Compatibility	3-214
2.9 Power Considerations	3-214
3.0 HARDWARE DESIGN CONSIDERATIONS	3-215
3.1 Hardware Upgradability	3-215
3.2 Hardware Decision Flowchart	3-215
4.0 SOFTWARE DESIGN AND UPGRADABILITY CONSIDERATIONS	3-216
4.1 Software Decision Flowchart	3-216
5.0 COMPATIBLE SOFTWARE ALGORITHM FLOWCHARTS	3-217
6.0 SUMMARY	3-220
7.0 REFERENCES	3-220

1.0 PURPOSE

The 28F016SA is the second member of Intel's FlashFile™ memory family. Its architecture evolved from that of the 28F008SA, Intel's first generation FlashFile memory device. The 28F016SA retains the standard 28F008SA's versatile capabilities and adds a Command Superset architecture which insures compatibility with the basic command-set.

This application note shows how to upgrade an existing 28F008SA-based design to the new 28F016SA memory device. Upgrades may require software modifications depending on the desired system functionality and straightforward hardware modifications to accommodate the new pinout.

2.0 SIMILARITIES AND DIFFERENCES BETWEEN THE 28F016SA AND THE 28F008SA

The 28F016SA memory is 100% command and algorithm, backward-compatible with the 28F008SA. It is defined as a Superset device which brings additional capabilities to system designs. Additional pins on the 28F016SA are added to define a user-selectable 8- or 16-bit wide memory, add on-chip write protection and multiple chip select signals. Note that you do not have to use the advanced features if you are performing a simple upgrade to the 28F016SA.

Before starting on your design upgrade, obtain the following specifications and application notes from Intel Corporation Literature Sales, 1 (800) 548-4725.

	Order #
28F008SA Data Sheet	290429
28F016SA Data Sheet	290489
28F016SA User's Manual	297372
28F008SA Software Drivers	292095
28F016SA Software Drivers	292126

2.1 Pinout Differences

Whereas the 28F008SA is 8-bit wide (40Ld-TSOP package), the 28F016SA is a high performance 16-bit wide flash file memory offering a user-configurable bus width (56Ld-TSOP package). Hence an additional 8 I/O pins are on the 28F016SA. Furthermore, the implementation of additional features such as Write Protect and Block Locking and user-selectable 3.3V and 5V operation require the definition of control pins for these functions. Finally, the optimization of the 28F016SA's architecture to achieve very high write performance resulted in a different pinout configuration from the 28F008SA.

Both device pinouts preserve the locations of I/O pins on the right-hand side and the sequence of pin functions of the 56Ld-TSOP package. However, it is still required to relay out an existing PCB design in order to accommodate the 16-Mbit chips. Table 1 lists all pin names and their numbers, highlighting all the changes.

28F008SA Pin Name	28F016SA Pin Name	28F008SA 40L-TSOP	28F016SA 56L-TSOP	Notes
A ₀	A ₀	24	32	
A ₁	A ₁	23	28	
A ₂	A ₂	22	27	
A ₃	A ₃	21	26	
A ₄	A ₄	20	25	
A ₅	A ₅	19	24	
A ₆	A ₆	18	23	
A ₇	A ₇	17	22	
A ₈	A ₈	16	20	
A ₉	A ₉	15	19	
A ₁₀	A ₁₀	14	18	
A ₁₁	A ₁₁	13	17	
A ₁₂	A ₁₂	8	13	
A ₁₃	A ₁₃	7	12	
A ₁₄	A ₁₄	6	11	
A ₁₅	A ₁₅	5	10	
A ₁₆	A ₁₆	4	8	
A ₁₇	A ₁₇	3	7	
A ₁₈	A ₁₈	2	6	
A ₁₉	A ₁₉	1	5	
—	A ₂₀	—	4	1
DQ ₀	DQ ₀	25	33	
DQ ₁	DQ ₁	26	35	
DQ ₂	DQ ₂	27	38	
DQ ₃	DQ ₃	28	40	
DQ ₄	DQ ₄	32	44	
DQ ₅	DQ ₅	33	46	

28F008SA Pin Name	28F016SA Pin Name	28F008SA 40L-TSOP	28F016SA 56L-TSOP	Notes
DQ ₆	DQ ₆	34	49	
DQ ₇	DQ ₇	35	51	
—	DQ ₈	—	34	
—	DQ ₉	—	36	
—	DQ ₁₀	—	39	
—	DQ ₁₁	—	41	
—	DQ ₁₂	—	45	
—	DQ ₁₃	—	47	
—	DQ ₁₄	—	50	
—	DQ ₁₅	—	52	
CE#	CE ₀ #	9	14	
—	CE ₁ #	—	2	2
RP#	RP#	12	16	3
RY/BY#	RY/BY#	36	53	4
OE#	OE#	37	54	
WE#	WE#	38	55	
—	BYTE#	—	31	5
—	WP#	—	56	
—	3/5#	—	1	6
V _{PP}	V _{PP}	11	15	
V _{SS}	V _{SS}	29, 30	21, 42, 48	
V _{CC}	V _{CC}	10, 31	9, 37, 43	
NC	NC	39, 40	3, 29, 30	

NOTES:

1. Highest Order Address
2. Dual CEx#
3. Formerly Called PWD#
4. Open Drain for 28F016SA
5. x8/x16 Selection
6. Selects Supply Voltage

2.1.1 HARDWARE COMPATIBLE CONFIGURATION

The following is an example which shows the state of all pins when operating in a 28F008SA-compatible mode:

WP# = V_{CC} (Write Protect feature disabled)

CE₀# = CE₁# (Chip Enable)

A₂₀ = GND or V_{CC} (selects upper/lower 1 MB)

3/5# = GND (5V operation) or V_{CC} (3.3V operation)

V_{CC} = 5.0V or 3.3V

BYTE# = GND (8-bit mode)

RY/BY# = Level Mode (set for default) with an external pull-up resistor

NOTE:

The 28F008SA has a CMOS driven RY/BY# output for interrupt capability.

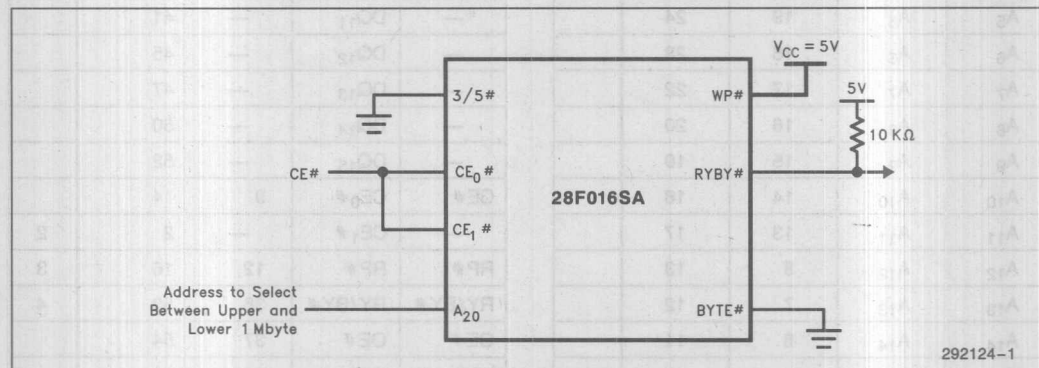


Figure 1. 28F016SA Configured as a 28F008SA-Compatible Memory

2.2 Compatible Command-Set

Byte Write	40H, 10H
Single Block Erase	20H
Erase Suspend to Read	B0H
Read Array	FFH
Read CSR	70H
Clear CSR	50H
Read Intelligent IDs	90H

Table 2. 28F008SA-Compatible Commands

Command	Notes	First Bus Cycle			Second Bus Cycle		
		Oper	Addr	Data	Oper	Addr	Data
Read Array		Write	X	FFH	Read	AA	AD
Intelligent Identifier	1	Write	X	90H	Read	IA	ID
Read Compatible Status Register	2	Write	X	70H	Read	X	CSRD
Clear Status Register	3	Write	X	50H			
Word/Byte Write		Write	X	40H	Write	WA	WD
Alternate Word/Byte Write		Write	X	10H	Write	WA	WD
Block Erase/Confirm		Write	X	20H	Write	BA	D0H
Erase Suspend/Resume		Write	X	B0H	Write	X	D0H

ADDRESS

AA = Array Address

BA = Block Address

IA = Identifier Address

WA = Write Address

X = Don't Care

DATA

AD = Array Data

CSR = Compatible Status Register

CSRD = CSR Data

GSR = Global Status Register

BSR = Block Status Register

ID = Identifier Data

WD = Write Data

NOTES:

- Following the Intelligent Identifier command, two Read operations access the manufacturer and device signature codes.
- The CSR is automatically available after device enters Data Write, Erase, or Suspend operations.
- Clears CSR.3, CSR.4 and CSR.5. Also clears GSR.5 and all BSR.5 and BSR.2 bits. See Status register definitions in the 28F016SA data sheet.

2.3 Technology Comparison

Both the 28F016SA and the 28F008SA are manufactured on Intel's Flash ETOX process technology. This technology is optimized for random access flash memory products with the highest read/write performance and lowest power consumption. The ETOX flash technology achieves very high reliability and quality.

2.4 Available Speeds

The 28F016SA designed on a 0.6 μm ETOX IV process, achieves faster speeds than the 28F008SA manufactured on 0.8 μm ETOX III process. Intel offers the 28F008SA in 5V-read version with speeds at 85/90 ns and 120 ns. The 28F016SA on the other hand is offered with a dual 3.3V and 5V read capability with speeds of (70/80 ns, 100 ns) and (120 ns, 150 ns) at 5V and 3.3V respectively. Note that both devices are offered at faster speeds (85 ns and 70 ns) under reduced loading conditions. *Please consult the data sheets referenced in this application note.*

2.5 Available Packages

The 28F016SA comes only in a 56Ld-Thin Small Outline Package (TSOP) optimized for its user-selectable x8/x16 memory architecture. The 28F008SA is offered in 2 packages, which are the 40Ld-TSOP (both standard and reverse pinout) and the 44Ld-Plastic Small Outline Package (PSOP).

2.6 Operating Modes

The 28F016SA behaves in the same manner as the 28F008SA. If a compatible command is written to the device, the Compatible Status Register contents are automatically put on the data bus. With the block locking disabled (WP# = high), the 28F016SA is identical to the 28F008SA, regardless of any lock-bit settings which define the lock state of a given block. The Command User Interface (CUI), Write State Machine (WSM) and Compatible Status Register (CSR) units function similarly on both devices.

The 28F016SA however, allows the user to issue multiple commands successively by watching the Queue bit (GSR.3 or BSR.3), a feature which improves performance, as described in the 28F016SA user's manual. Consult the 28F016SA data sheet and 28F016SA user's manual for detail operation.

2.7 AC Compatibility

The 28F016SA specifies the output Load circuit as an equivalent transmission line model which reflects the timing delays more accurately. The same diode/resistor circuit combination found in the 28F008SA data sheet also applies to the 28F016SA.

Address and Data are latched on the rising edge of WE# for both devices. All AC timing specifications are similar except for the differences noted in the data sheets. The 28F016SA has additional write timings describing the on-chip page buffers which do not exist on the 28F008SA.

2.8 DC Compatibility

Whereas the 28F008SA only operates at 5V V_{CC} and the 28F008SA-L only operates at 3.3V V_{CC} , the 28F016SA operates at both 3.3V and 5V V_{CC} supply voltages. In 5V mode of operation, the two devices are similar in terms of input/output level specifications. Consult the data sheets for differences in current consumption.

2.9 Power Considerations

In addition to the active, standby and deep power-down modes which exist on the 28F008SA, the 28F016SA has additional current modes useful in power management applications.

The two modes are:

Automatic Power Saving feature which is activated whenever the device addresses are not switching which is equivalent to a static mode of operation on the chip is accessed by a slowed clock. In this state, the chip typically draws less than 1 mA of total current.

The second low current mode of operation is enabled through command control, using the Sleep command. A Sleep command written to the device will put the chip in the lowest current state (deep power-down level) after all active operations have been processed. The device retains the value of its status registers while in sleep mode.

Note that the AC and DC specifications mentioned here are valid at the time this application note was written. Please reference the device data sheets for the latest up-to-date specifications.

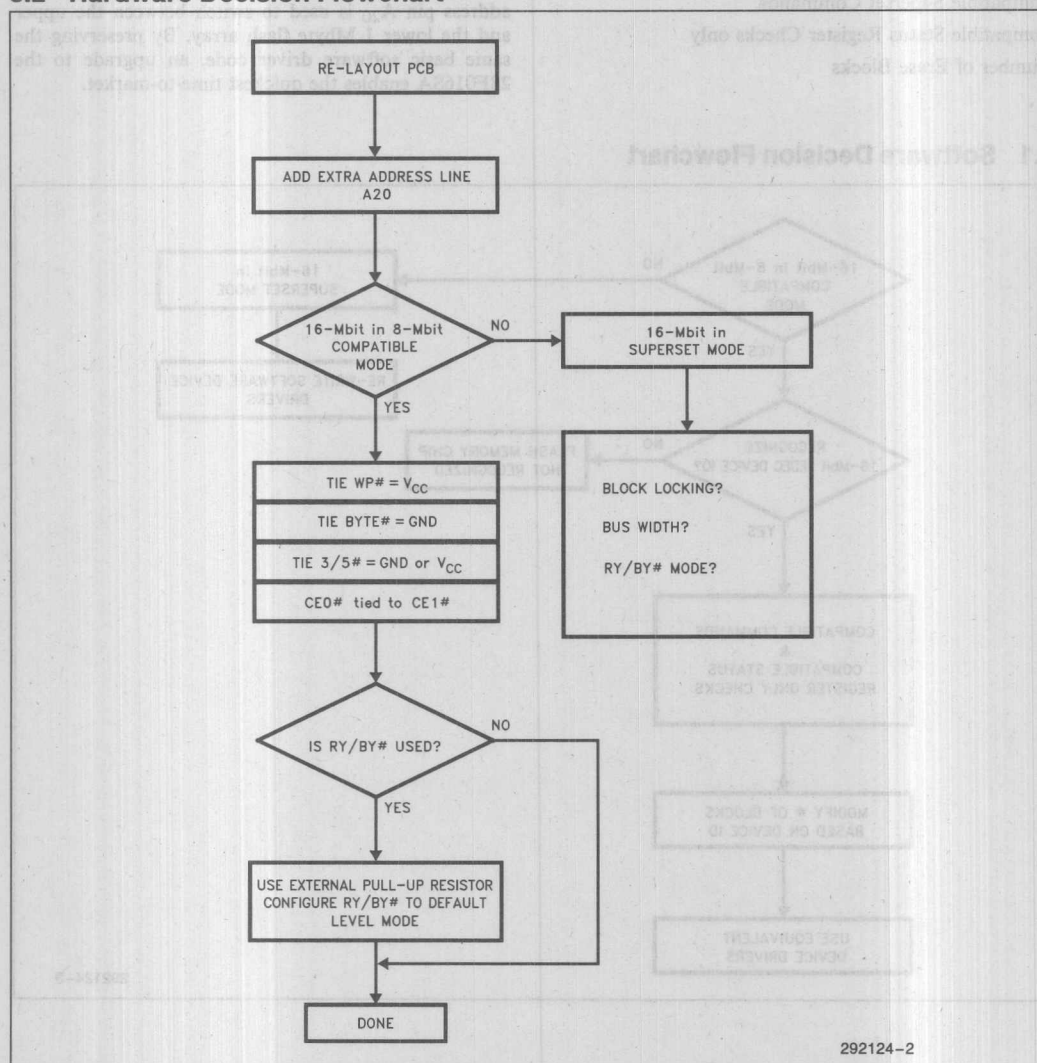
3.0 HARDWARE DESIGN CONSIDERATIONS

If you are considering a density upgrade to the 28F016SA, careful attention to certain areas must be followed. This section is not intended to cover all potential issues related to system design, but rather as a guideline in designing an upgrade to the 28F016SA.

3.1 Hardware Upgradability

The following flowchart summarizes the logical steps a system designer must go through to complete a density upgrade. This simple upgradability procedure allows the 28F016SA-based system to achieve quicker time-to-market while incorporating future expandability to take advantage of the Superset features of the 28F016SA.

3.2 Hardware Decision Flowchart



4.0 SOFTWARE DESIGN AND UPGRADABILITY CONSIDERATIONS

In order to do a software upgrade to the 28F016SA, the software designer must pay attention to a few key areas. They can be grouped as follows:

Device Intelligent Identifier = A0H (versus A2H for 28F008SA)

Compatible Superset Commands

Compatible Status Register Checks only

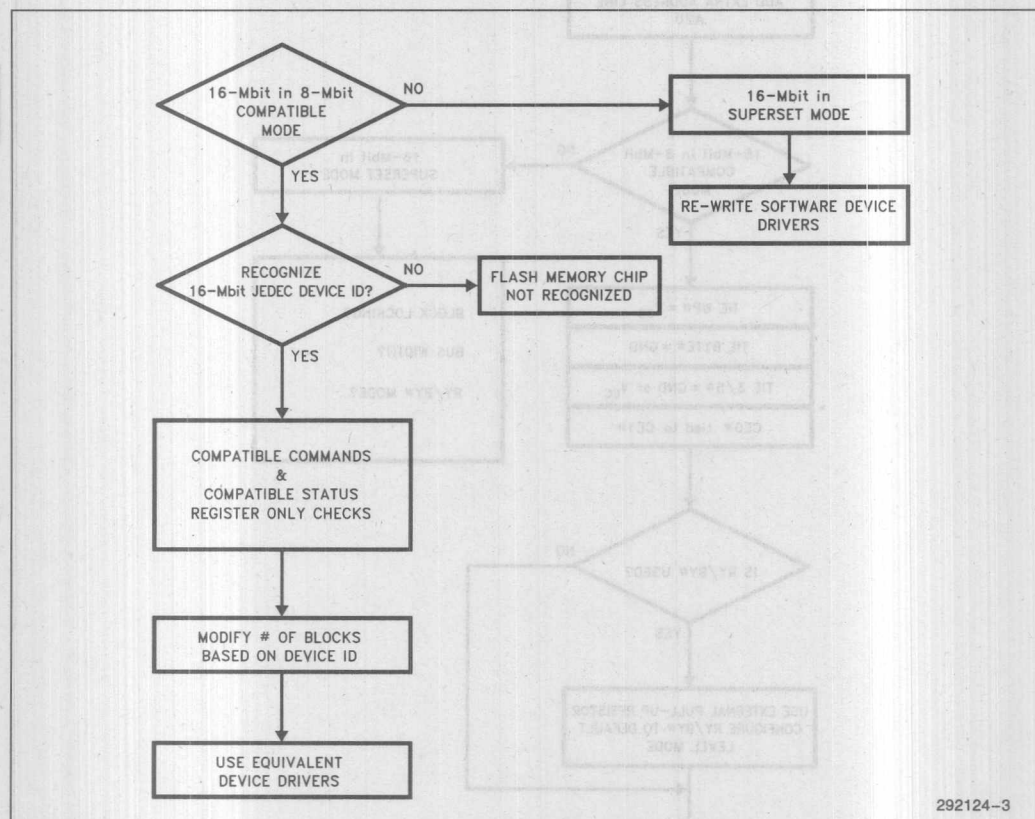
Number of Erase Blocks

Device Drivers for the 28F008SA and the 28F016SA are provided in application notes AP-360 and AP-377 respectively.

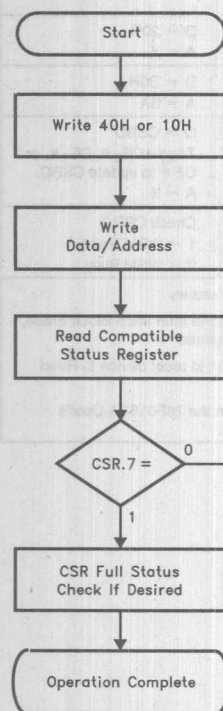
Software drivers written for the 28F008SA need to recognize the new device ID and change the memory size boundaries in order to work on a 28F016SA-based system design.

Note that the 28F016SA can be treated as two 8-Mbit memory devices in a single package. The highest order address pin A₂₀ is used to switch between the upper and the lower 1 Mbyte flash array. By preserving the same basic software driver code, an upgrade to the 28F016SA enables the quickest time-to-market.

4.1 Software Decision Flowchart

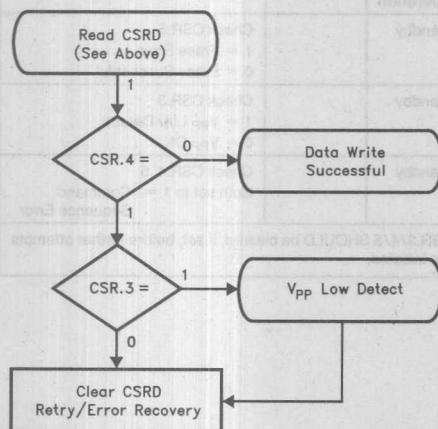


5.0 COMPATIBLE SOFTWARE ALGORITHM FLOWCHARTS



292124-4

CSR FULL STATUS CHECK PROCEDURE



292124-5

Bus Operation	Command	Comments
Write	Word/Byte Write	D = 40H or 10H A = X
Write		D = WD A = WA
Read		Q = CSRD Toggle CE ₀ #, CE ₁ #, or OE# to update CSRD. A = X
Standby		Check CSR.7 1 = WSM Ready 0 = WSM Busy

Repeat for subsequent Word/Byte Writes.
 CSR Full Status Check can be done after each Word/Byte Write, or after a sequence of Word/Byte Writes.
 Write FFH after the last operation to reset device to Read Array Mode.
 See Command Bus definitions in the 28F016SA User's Manual for description of codes.

3

Bus Operation	Command	Comments
Standby		Check CSR.4 1 = Data Write Unsuccessful 0 = Data Write Successful
Standby		Check CSR.3 1 = V _{pp} Low Detect 0 = V _{pp} OK

CSR.3/4 SHOULD be cleared, if set, before further attempts are initiated.

Figure 2. Word/Byte Writes

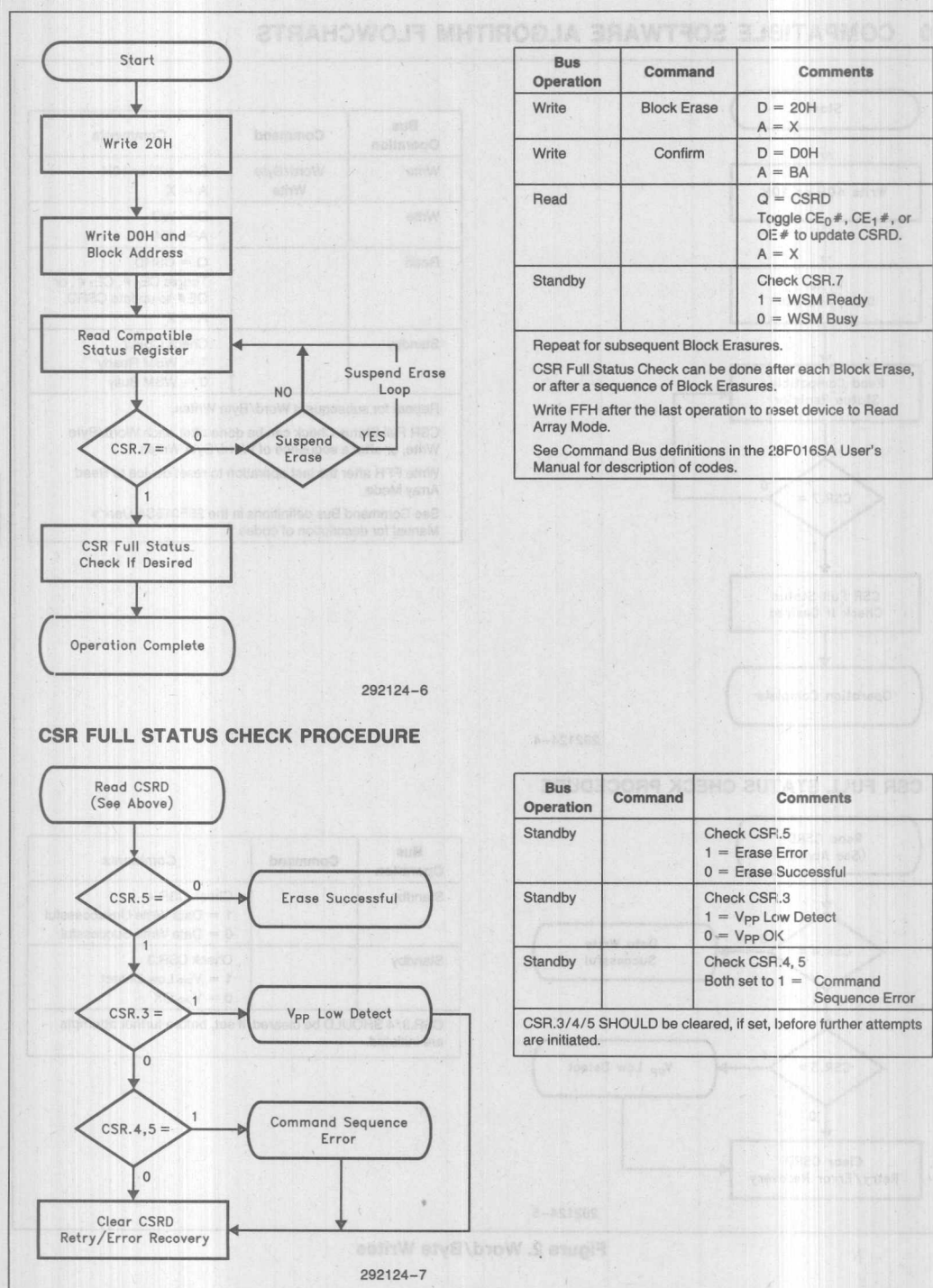
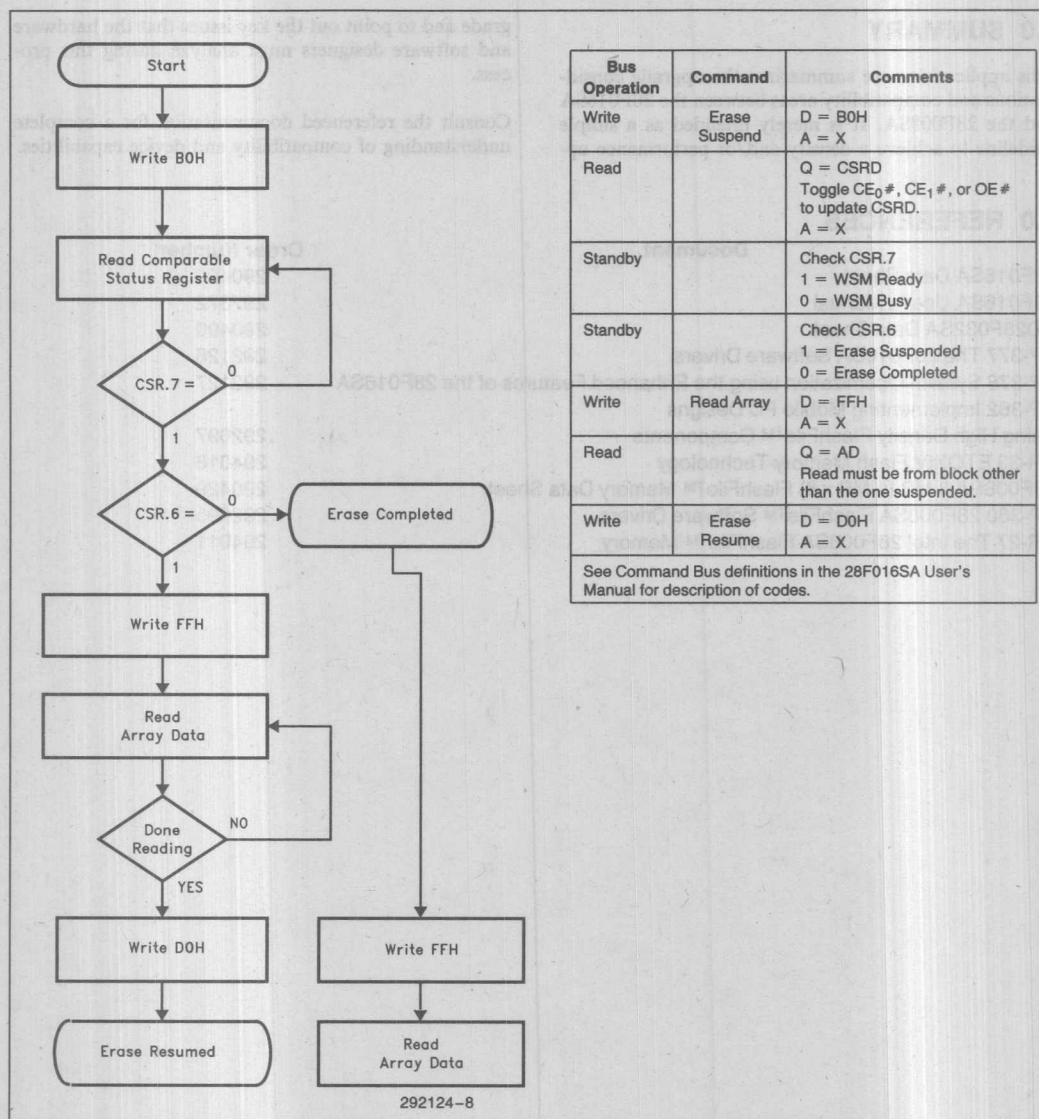


Figure 3. Block Erase



3

Figure 4. Erase Suspend to Read Array

6.0 SUMMARY

This application note summarizes the upgrade considerations and compatibility areas between the 28F016SA and the 28F008SA. It is merely intended as a simple guideline to achieve a density and/or performance up-

grade and to point out the key issues that the hardware and software designers must analyze during this process.

Consult the referenced documentation for a complete understanding of compatibility and device capabilities.

7.0 REFERENCES

Document	Order Number
28F016SA Data Sheet	290489
28F016SA User's Manual	297372
DD28F032SA Data Sheet	290490
AP-377 The 28F016SA Software Drivers	292126
AP-378 System Optimization using the Enhanced Features of the 28F016SA	292127
AP-362 Implementing Mobile PC Designs Using High Density FlashFile™ Components	292097
ER-33 ETOXIV Flash Memory Technology	294016
28F008SA 8 MB (1 MB x 8) FlashFile™ Memory Data Sheet	290429
AP-360 28F008SA FlashFile™ Software Drivers	292095
ER-27 The Intel 28F008SA FlashFile™ Memory	294011

APPLICATION NOTE

28F016SA SOFTWARE DRIVERS

3

PATRICK KILLELEA
MCD APPLICATIONS ENGINEERING

SALIM FEDEL
MCD APPLICATIONS ENGINEERING

October 1993

Order Number: 292126-001

3-221

28F016SA SOFTWARE DRIVERS

CONTENTS

PAGE

INTRODUCTION	3-223
28F016SA C DRIVERS	3-226
28F016SA ASM86 DRIVERS	3-241
GLOSSARY OF TERMS	3-258
ADDITIONAL INFORMATION	3-258

INTRODUCTION

ABOUT THE CODE

This application note provides example software code for word writing, block erasing, and otherwise controlling Intel's 28F016SA 16 Mbit symmetrically blocked FlashFile™ memory. Two programming languages are provided: high-level "C" for broad platform support, and more optimized ASM86 assembly. In many cases, the driver routines can be inserted "as is" into the main body of code being developed by the system software engineer. Extensive comments are included in each routine to facilitate adapting the code to specific applications.

The internal automation of the 28F016SA makes software timing loops unnecessary and results in platform-independent code. The following example code is designed to be executed in any type of memory and with all processor clock rates. "C" code can be used with many microprocessors and microcontrollers, while ASM86 assembly code provides a solution optimized for Intel microprocessors and embedded processors.

The 28F016SA, like the 28F008SA, is divided into 64 Kbyte blocks. Since the GSR and BSR are defined relative to the nearest preceding block beginning address, I often refer to this "block base" address in the comments.

Assumptions:

- Pointers (in C) or EDI offsets (in ASM86) are 4 bytes long, providing a flat addressing space over the entire 28F016SA device. This implies the use of 386 or higher machines. If the code is to be run on a machine with a smaller address space, the code must be modified to include some sort of "windowing" scheme which maps segments of flash into system memory. The Intel 82365 is commonly used for this purpose.
- "Ints" are 16-bit and "longs" 32-bit in C.
- There exists a function which can toggle an individual 28F016SA pin, given the pin number.
- The C code can access a function which derives the corresponding block base address from any given address.
- BYTE# pin on the device determines whether addressing refers to words or bytes. I assume word writes/reads to a single device. The code should be valid, however, for pairs 28F016SAs in byte mode.

- 28F016SA commands can be written to any address in the block or device to be affected by that command. Note that a word-long command written to the last byte in a block will overlap into the first byte of the next block.

Both the C and ASM86 code in this document contain the following routines, in this order:

compatible_block_erase (compatible with 28F008SA)
compatible_suspend_to_read (compatible with 28F008SA)
compatible_byte_write (compatible with 28F008SA)
ESR_block_erase
ESR_status_check_after_erase
ESR_status_check_after_write
ESR_suspend_to_read
ESR_word_write
erase_all_unlocked_blocks
lock_block
status_upload
pagebuffer_write_to_flash
sequential_pagebuffer_load
single_pagebuffer_load
two_byte_write
write_during_erase

ABOUT THE 28F016SA

Companion product datasheets for the 28F016SA should be reviewed in conjunction with this application note for a complete understanding of the device.

The example code makes extensive use of bit-masking when interpreting the status registers. As a quick review, note that any bit in a register can be tested by bitwise ANDing the register with the appropriate power of two. Since all of the bits other than the one being tested are masked out, testing the resulting byte for truth is the same as testing the desired bit for truth. For example, if a register contains 01001010, the test for bit 3 would be ANDing the register with 00001000, or hex 8, and testing the result for truth:

Binary	Hex	Register
01001010	4A	Mask for bit 3
& 00001000	& 08	Result
= 00001000	= 08	

In this case the result byte is true, indicating that bit 3 in the register was a 1.

The meanings of the individual bits of these registers are presented here for reference. Note that there are two status register spaces, both of which are distinct from the flash memory array address space. In the CSR space, the CSR is mapped to every address. In the ESR space, the GSR is mapped two words above the base of each 64 Kbyte block, i.e. to addresses 2, 8002H, 10002H, etc. (in word mode), while each BSR is similarly mapped one word above the base of each 64 Kbyte block to locations 1, 8001H, 10001H, etc. (in word mode), each BSR reflecting the status of its own block.

CSR.7	Write State Machine Status	1 = ready 0 = busy
CSR.6	Erase-suspend Status	1 = erase suspended 0 = erase in progress/ completed
CSR.5	Erase Status	1 = error in block erase 0 = successful block erase
CSR.4	Data-write Status	1 = error in data write 0 = successful data write
CSR.3	V _{pp} Status	1 = V _{pp} low detect/ operation aborted 0 = V _{pp} OK when operation occurred
CSR.2	Reserved for future use	
CSR.1	Reserved for future use	
CSR.0	Reserved for future use	

GSR.7	Write State Machine Status	1 = ready 0 = busy
GSR.6	Operation-suspend Status	1 = operation suspended 0 = operation in progress/ completed
GSR.5	Device Operation Status	1 = operation unsuccessful 0 = operation successful or running
GSR.4	Device Sleep Status	1 = device in sleep 0 = device not in sleep
GSR.3	Queue Status	1 = queue full 0 = queue available
GSR.2	Page Buffer Availability	1 = one/two page buffers available 0 = no page buffers available
GSR.1	Page Buffer Status	1 = selected page buffer ready 0 = selected page buffer busy
GSR.0	Page Buffer Select Status	1 = page buffer 1 selected 0 = page buffer 0 selected

BSR.7	Block Status	1 = ready 0 = busy
BSR.6	Block-lock Status	1 = block unlocked for write/erase 0 = block locked to write/erase
BSR.5	Block Operation Status	1 = error in block operation 0 = successful block operation
BSR.4	Block Operation Abort Status	1 = block operation aborted 0 = block operation not aborted
BSR.3	Queue Status	1 = device queue full 0 = device queue available
BSR.2	V _{pp} Status	1 = V _{pp} low detected 0 = V _{pp} OK when operation occurred
BSR.1	Reserved for future use	
BSR.0	Reserved for future use	

28F016SA Commands

The 28F016SA command set is a superset of the 28F008SA command set, giving existing 28F008SA code the ability to run on the 28F016SA with minimal modifications.

28F008SA-Compatible Commands

00	invalid/reserved
20	single block erase
40	word/byte write
50	clear status registers
70	read CSR
90	read ID codes
B0	erase suspend
D0	confirm/resume
FF	read flash array

28F016SA Performance-Enhancement Commands

0C	page buffer write to flash
71	read GSR and BSRs (i.e. the ESR)
72	page buffer swap
74	single load to page buffer
75	read page buffer
77	lock block
80	abort
96,01	RY/BY# enable to level-mode
96,02	RY/BY# pulse on write
96,03	RY/BY# pulse on erase
96,04	RY/BY# pin disable
97	upload BSRs with lock bit
99	upload device information
A7	erase all unlocked blocks
E0	sequential load to page buffer
F0	sleep
FB	two-byte write

"C" DRIVERS

```

/*****
/* Copyright Intel Corporation, 1993
/* Example C Routines for the 28F016SA Flash memory component
/* Patrick Killelea, Intel Corporation
/* Revision 1.0, 25 June 1993
/*
/* NOTE: BYTE# pin on the device determines whether addressing
/* refers to words or bytes. I assume word mode.
/* NOTE: A 28F016SA command can be written to any address in the
/* block or device to be affected by that command.
*****/

#include <stdio.h>

void set_pin(int pin, int level)
{
/* set_pin is an implementation-dependent function which sets a
/* given pin on the standard 28F016SA pinout HIGH = 1 or LOW = 0.
}

int *base(int*address)
{
/* base is an implementation-dependent function which takes an
/* address in the flash array and returns a pointer to the base
/* of that 64 Kbyte block.
}

```



```

int compatible_block_erase(int*address)
{
/* This procedure erases a 64 Kbyte block on the 28F016SA. */
/* It also works with a pair of 28F008SAs. */
int CSR;
/* CSR variable is used to return contents of CSR register. */

*address = 0X2020;
/* Single Block Erase command
*address = 0XD0D0;
/* Confirm command
while(!(0X0080 & *address))
/* Poll CSR until CSR.7 = 1 (WSM ready)
{
/* Erase may be suspended here to write to a different block.
};
/* At this point, CSR.7 is 1, indicating WSM is not busy.
/* Note that we are still reading from CSR by default.
CSR = *address;
/* Save CSR before clearing it.
*address = 0X5050;
/* Clear Status Registers command
return(CSR);
/* Return CSR to be checked for status of operation.
}

```

```

int compatible_byte_write(int address, int data)
{
    /* This procedure writes a byte to the 28F016SA. */
    /* It also works with the 28F008SA. */
    int CSR;
    /* CSR variable is used to return contents of CSR register. */

    *address = 0X1010;
    /* Word Write command
    *address = data;
    /* Actual data write to flash address.
    while(!(0X0080 & *address));
    /* Poll CSR until CSR.7 = 1 (WSM ready)

    CSR = *address;
    /* Save CSR before clearing it.
    *address = 0X5050;
    /* Clear Status Registers command
    return(CSR);
    /* Return CSR to be checked for status of operation.
    }

```

```

/* This procedure suspends an erase operation to do a read. */
/* It also works with a pair of 28F008SAs. */
int CSR;
/* CSR variable is used to return contents of CSR register. */

/* Assume erase is underway in block beginning at erase_address. */
*erase_address = OXB0B0;
/* Erase Suspend command */
while(!(OX0080 & *erase_address));
/* Poll CSR until CSR.7 = 1 (WSM ready) */
*erase_address = OXFFFF;
/* Read Flash Array command */
*result = *read_address;
/* Do the actual read. Any number of reads can be done here. */
if (OX0040 & *erase_address)
/* If CSR.6 = 1 (erase incomplete)
    *erase_address = OXD0D0;
/* Erase Resume command
CSR = *erase_address;
/* Save CSR before clearing it.
*erase_address = OX5050;
/* Clear Status Registers command
return(CSR);
/* Return CSR to be checked for status of operation.
}

```

```

int device_information_upload(int*address)
/*This procedure uploads the device revision number to the variable GSR_DSR.*/
{
    int GSR_DSR;
    /*DSR variable is used to return device revision status.*/
    int *block_base = base(address);
    /*Find pointer to base of 32K word block.*/

    *address = 0X7171;
    /*Read Extended Status Registers command
    while (0X0008 & *(block_base + 2));
    /*Poll GSR until GSR.3 = 0 (queue available).
    while (!(0X0080 & *(block_base + 2)));
    /*Poll GSR until GSR.7 = 1 (WSM available).
    *address = 0X9999;
    /*Device information Upload command
    *address = 0XD0D0;
    /*Confirmation command
    *address = 0X7171;
    /*Read Extended Status Registers command
    while (!(0X0080 & *(block_base + 2)));
    /*Poll GSR until GSR.7 = 1 (WSM not busy)
    *address = 0X7272;
    /*Swap page buffer to bring buffer with status information to top.
    *address = 0X7575;
    /*Read Page Buffer command
    GSR_DSR = (*(block_base + 3) & 0X00FF);
    /*Put device revision code in bottom byte of return value.
    /*Note that device revision code was read from word 3 in page buffer.
    *address = 0X7171;
    /*Read Extended Status Registers command
    GSR_DSR += (*(block_base + 2) & 0XFF00);
    /*Put GSR in top byte of return value.
    /*User should check GSR for operation success
    *address = 0X5050;
    /*Clear Status Registers command
    return(GSR_DSR);
}

```



```

int ESR_block_erase(int*erase_address)
/* This procedure erases a block on the 28F016SA. */
{
    int ESR;
    /* ESR variable is used to return contents of GSR and BSR. */
    int*block_base = base(erase_address);
    /* Find address of base of block being erased. */
    *erase_address = 0X7171;
    /* Read Extended Status Registers command */
    while (0X0008 & *(block_base + 1));
    /* Poll BSR until BSR.3 of erase_address = 0 (queue available). */
    /* BSR is 1 word above base of target block in ESR space. */
    *erase_address = 0X2020;
    /* Single Block Erase command */
    *erase_address = 0XD0D0;
    /* Confirm command */
    *erase_address = 0X7171;
    /* Read Extended Status Registers command */
    while (!(0X0080 & *(block_base + 1)));
    /* Poll BSR until BSR.7 of target erase_address = 1 (block ready). */
    ESR = (*(block_base + 2) & 0XFF00) + (*(block_base + 1) & 0X00FF);
    /* Put GSR in top byte and BSR in bottom byte of return value. */
    *erase_address = 0X5050;
    /* Clear Status Registers command */
    return(ESR);
}

```

```

int ESR_suspend_to_read(int*address,int*result)
/* This procedure suspends an erase on the 28F016SA. */
{
/* Address is assumed to point to location to be read. */
/* result is used to hold read value until procedure is complete. */
int ESR;
/* ESR variable is used to return contents of GSR and BSR. */
int*block_base = base(address);

*address = 0X7171;
/* Read Extended Status Registers command */
while (!(0X0080 & *(block_base + 1)));
/* Poll BSR until BSR.7 of target address = 1 (block ready). */
/* BSR is 1 word above base of target block in ESR space. */
*address = 0XB0B0;
/* Operation Suspend command */
*address = 0X7171;
/* Read Extended Status Registers command */
while (!(0X0080 & *(block_base + 2)));
/* Poll GSR until GSR.7 = 1 (WSM ready). */
*address = 0xFFFF;
/* Read Flash Array command */
*result = *address;
/* Read the data. */
if (0X0040 & *(block_base + 2))
/* If GSR.6 indicates an operation was suspended on this device, */
*address = 0XD0D0;
/* then resume the operation. */

ESR = (*(block_base + 2) & 0XFF00) + (*(block_base + 1) & 0X00FF);
/* Put GSR in top byte and BSR in bottom byte of return value. */
*address = 0X5050;
/* Clear Status Registers command */
return(ESR);
}

```

```

{
int ESR;
/* ESR variable is used to return contents of GSR and BSR. */
int*block_base = base(write_address);

*write_address = 0X7171;
/* Read Extended Status Registers command */
while (0X0008 & *(block_base + 1));
/* Poll BSR until BSR.3 of target address = 0 (queue available). */
/* BSR is 1 word above base of target block in status reg space. */
*write_address = 0X1010;
/* Write word command */
*write_address = data;
/* Write actual data. */
*write_address = 0X7171;
/* Read Extended Status Registers command */
while (!(0X0080 & *(block_base + 1)));
/* Poll BSR until BSR.7 of target address = 1 (block ready). */
ESR = (*(block_base + 2) & 0XFF00) + (*(block_base + 1) & 0X00FF);
/* Put GSR in top byte and BSR in bottom byte of return value. */
*write_address = 0X5050;
/* Clear Status Registers command */
return(ESR);
}

```

```

{
int GSR;
/* Return value will contain GSR in both top and bottom byte. */
/* 32 bit long pointed to by failure_list is used to return map */
/* of block failures, each bit representing one block's status. */
/* device_address points to base of chip. */
int block;
/* block is used to hold block count for loop through blocks. */
long power = 1;

*failure_list = 0;
/* Initialize all 32 bits of failure list long to 0. */
*device_address = 0X7171;
/* Read Extended Status Registers
while (0X0008 & *(device_address + 1));
/* Poll BSR until BSR.3 of target address = 0 (queue available). */
*device_address = 0XA7A7;
/* Full-chip erase command
*device_address = 0XD0D0;
/* Confirm command
*device_address = 0X7171;
/* Read Extended Status Registers command
while (!(0X0080 & *(device_address + 2)));
/* Poll GSR until GSR.7 = 1 (WSM ready)

for (block = 0; block < 0X0020; block++)
{
/* Go through blocks, looking at each BSR.5 for operation failure */
/* and setting appropriate bit in long pointed to by failure list. */
if (0X0020 & *(device_address + block * 0X8000 + 1))
/* Multiply block by 32K words to get to the base of each block. */
*failure_list += power;
/* If the block failed, set that bit in the failure list. */
power *= 2;
/* Increment to next power of two to access next bit.

GSR = *(device_address + 2);
/* Put GSR in both bytes of return value.
*device_address = 0X5050;
/* Clear Status Registers command
return(GSR);
}

```



```

int lock_block(int*lock_address)
/* This procedure locks a block on the 28F016SA. */
{
    int ESR;
    /* ESR variable is used to return contents of GSR and BSR. */
    #define WPB 56
    /* Write Protect pin (active low) is pin number 56 on standard */
    /* pinout of 28F016SA. */
    #define VPP 15
    /* Vpp pin is pin number 15 on standard pinout of 28F016SA. */
    #define HIGH 1
    #define LOW 0

    int*block_base = base(lock_address);
    /* Find pointer to base of block being locked.

    *lock_address = 0X7171;
    /* Read Extended Status Registers command
    while (0X0008 & *(block_base + 2));
    /* Poll GSR until GSR.3 = 0 (queue available).
    set_pin(WPB, HIGH);
    /* Disable write protection by setting WPB high.
    set_pin(VPP, HIGH);
    /* Enable Vpp, wait for ramp if necessary in this system.
    *lock_address = 0X7777;
    /* Lock Block command
    *lock_address = 0XD0D0;
    /* Confirmation command
    *lock_address = 0X7171;
    /* Read Extended Status Registers command
    while (!(0X0080 & *(block_base + 2)));
    /* GSR is 2 words above 0; poll GSR until GSR.7 = 1 (WSM ready).

    ESR = (*(block_base + 2) & 0XFF00) + (*(block_base + 1) & 0X00FF);
    /* Put GSR in top byte and BSR in bottom byte of return value.
    *lock_address = 0X5050;
    /* Clear Status Registers command
    return(ESR);
}

```

```

int status_upload(int*address)
/* This procedure uploads status information into the ESR from non-volatile */
/* status bits. */
{
    int ESR;
    /* ESR variable is used to return contents of GSR and BSR. */
    int*block_base = base(address);
    /* Find pointer to base of 32K word block.

    *address = 0X7171;
    /* Read Extended Status Registers command
    while (0X0008 & *(block_base + 2));
    /* Poll GSR until GSR.3 = 1 (queue available).
    *address = 0X9797;
    /* Lock-status Upload command
    *address = 0XD0D0;
    /* Confirmation command
    *address = 0X7171;
    /* Read Extended Status Registers command
    while (!(0X0080 & *(block_base + 2)));
    /* Poll GSR until GSR.7 = 1 (WSM not busy)

    ESR = (*(block_base + 2) & 0XFF00) + (*(block_base + 1) & 0X00FF);
    /* Put GSR in top byte and BSR in bottom byte of return value.
    *address = 0X5050;
    /* Clear Status Registers command
    return(ESR);
}

```

```

/* This routine assumes page buffer is already loaded. */
/* Address is where in flash array to begin writing. */
/* Low byte of byte_count must be 256 or fewer, high must be 0. */
/* High byte exists for future Page Buffer expandability. */
int ESR;
/* ESR variable is used to return contents of GSR and BSR. */
int*block_base = base(address);
/* Find pointer to base of block to be written. */

*address = 0X7171;
/* Read Extended Status Registers command
while (0X0008 & *(block_base + 1));
/* Poll BSR until BSR.3 of target address = 0 (queue available).
*address = 0X0C0C;
/* Page Buffer Write to Flash command
*address = byte_count;
/* Only A0 valid here; low or high byte loaded depending on A0.
*address = byte_count;
/* A0 internally complemented; alternate byte loads; write starts
*address = 0X7171;
/* Read Extended Status Registers command
while (!(0X0080 & *(block_base + 1)));
/* Poll BSR until BSR.7 of target address = 1 (block ready).

ESR = (*(block_base + 2) & 0XFF00) + (*(block_base + 1) & 0X00FF);
/* Put GSR in top byte and BSR in bottom byte of return value.
*address = 0X5050;
/* Clear Status Registers command
return(ESR);
}

```

```

void sequential_pagebuffer_load(int*device_address, char*start_address,
int word_count, int* data)
/* This procedure loads a multiple bytes to a page buffer. */
{
/* Low byte of word_count must be 128 or fewer, high must be 0. */
/* High byte exists for future Page Buffer expandability. */
char counter;
/* counter is used to keep track of bytes written. */

*device_address = 0X7171;
/* Read Extended Status Registers command */
while (!(0X0004 & *(device_address + 2)));
/* Poll GSR until GSR.2 = 1 (page buffer available). */
*device_address = 0XE0E0;
/* Sequential Page Buffer Load command */
*start_address = word_count;
/* Loads high or low byte of count register, depending on A0 */
*start_address = word_count;
/* Automatically loads alternate byte of count register
for (counter = 0; counter < word_count; counter++)
    *(start_address + counter) = data[counter];
/* Loop through data, writing to page buffer. */
/* This routine does not affect status registers.
}

```



```

void single_pagebuffer_load(int*device_address, char*address, int data)
/* This procedure loads a single byte or word to a page buffer. */
{
*device_address = 0X7171;
/* Read Extended Status Registers command */
while (!(0X0004 & *(device_address + 2)));
/* Poll GSR until GSR.2 = 1 (page buffer available) */
*device_address = 0X7474;
/* Single Load to Page Buffer command */
*address = data;
/* Actual write to page buffer */
/* This routine does not affect status registers. */
}

int two_byte_write(int*address, char data_low, char data_high)
/* This routine is used when BYTE# is low, i.e. the 28F016SA */
/* is in byte mode, to emulate a word write. */
{
int ESR;
/* ESR variable is used to return contents of GSR and BSR. */
int*block_base = base(address);
/* Find pointer to base of block. */

*address = 0X7171;
/* Read Extended Status Registers command */
while (0X0008 & *(block_base + 1));
/* Poll BSR until BSR.3 of target address = 0 (queue available). */
*address = 0XFBBF;
/* Two-byte Write command */
*address = data_low;
/* Load one byte of data register; A0 = 0 loads low byte, A1 high */
*address = data_high;
/* 28F016SA automatically loads alternate byte of data register */

/* Write is initiated. Now we poll for successful completion. */
*address = 0X7171;
/* Read Extended Status Registers command */
while (!(0X0080 & *(block_base + 1)));
/* Poll BSR until BSR.7 of target address = 1 (block ready). */
/* BSR is 1 word above base of target block in status reg space. */

ESR = (*(block_base + 2) & 0XFF00) + (*(block_base + 1) & 0X00FF);
/* Put GSR in top byte and BSR in bottom byte of return value. */
*address = 0X5050;
/* Clear Status Registers command */
return(ESR);
}

```

```

/* This procedure writes to one block while another is erasing. */
{
int ESR;
/* ESR variable is used to return contents of GSR and BSR. */
int*block_base = base(erase_address);
/* Find pointer to base of block being erased. */

*erase_address = 0X7171;
/* Read Extended Status Register command
while (0X0008 & *(block_base + 1));
/* Poll BSR until BSR.3 of target address = 0 (queue available).
/* BSR is 1 word above base of target block in ESR space.
*erase_address = 0X2020;
/* Erase Block command
*erase_address = 0XD0D0;
/* Confirm command
*write_address = 0X4040;
/* Word Write command
*write_address = data;
/* Write actual data.
/* Erase suspends, write takes place, then erase resumes.
*erase_address = 0X7171;
/* Read Extended Status Registers command
while (!(0X0080 & *(block_base + 1)));
/* Poll BSR until BSR.7 of erase address = 1 (block ready).
/* BSR is 1 word above base of target block in status reg space.

ESR = (*(block_base + 2) & 0XFF00) + (*(block_base + 1) & 0X00FF);
/* Put GSR in top byte and BSR in bottom byte of return value.
*block_base = 0X5050;
/* Clear Status Registers command
return(ESR);
}

```

```

; Copyright Intel Corporation, 1993
; EXAMPLE ASM86 Drivers for the 28F016SA Flash memory component
; Patrick Killelea, Intel Corporation
; Revision 1.0, August 6, 1993
; NOTE:
; The code assumes 32-bit flat model protected mode for simplicity.
; i.e. ES contains 0 and EDI accesses the entire memory space.

```

```

TEXT    segment byte    public 'CODE'
        assume  cs:TEXT

```

```

; Following is the structure by which all parameters are passed.

```

```

params STRUCT

```

```

        erase_addr    DD    ?    ; base of block or device to erase
        write_addr    DD    ?    ; address to write to
        write_base    DD    ?    ; base address of block written to
        read_addr     DD    ?    ; address to read from
        read_base     DD    ?    ; base address of block read from
        lock_addr     DD    ?    ; base address of block to lock
        data_addr     DD    ?    ; address of data to write
        data          DW    ?    ; data word to write
        pagebuffer_start_addr DB ?    ; start address in page buffer
        pagebuffer_word_count DW ?    ; number of words for pb
                                           ; read/write

```

```

params ENDS

```

```

; MACRO    set_pin
; This macro pushes parameters needed for the set_pin routine, calls
; set_pin, and then pops those parameters. set_pin is an implementation-
; dependent function which sets a given pin on the standard 28F016SA
; pinout HIGH = 1 or LOW = 0.
;
; Data needed at the beginning of this macro:
; pin:      28F016SA pin number
; level:    level to set pin
; Trashes: CX

```

```

MACRO    set_pin    pin, level
        push    pin                ; Push pin number of Write Protect#
        push    level             ; Push logic level of pin
        call    near ptr set_pin  ; Call set_pin
        pop     CX                ; Pop off parameters
        pop     CX

```

```

ENDM

```

```

;=====
; PROCEDURE      compatible_block_erase
; This procedure erases a 64 Kbyte block on the 28F016SA.
; It also works with a pair of 28F008SAs.
; Param fields needed:
;
;      erase_address: offset of base of 28F016SA block to erase
; Output  BX: CSR, duplicated in both high and low bytes
;=====
compatible_block_erase    proc    near
    mov     EDI,params.erase_addr
    mov     ES:[EDI],2020H          ; Block Erase command
    mov     ES:[EDI],D0D0H          ; Erase Confirm command
; Note that it is not strictly necessary to write an erase command to
; the base of a block; any address within the block will do.

WSM_not_ready2:
    mov     AX,ES:[EDI]             ; Read CSR.
    test    AX,80H                  ; If CSR.7 = 0, test sets ZF.
    ; Erase may be suspended here to write to a different block.
    jz      short WSM_not_ready2    ; Loop while ZF is set.
    ; Poll CSR until CSR.7 = 1, indicating that WSM is ready.

    mov     BX,AX                   ; Return CSR in BX.
    mov     ES:[EDI],5050H          ; Clear Status Registers command

    ret                             ; Return to calling routine.
compatible_block_erase    endp

```



```

;=====
; PROCEDURE      compatible_byte_write
; This procedure writes a byte to the 28F016SA. It also works with the
; 28F008SA.
; Param fields needed:
;      params.data: data word to be written
;      params.write_addr: offset of 28F016SA address to write
; Output:  BX: CSR, duplicated in both high and low bytes
;=====

```

```

compatible_byte_write proc near
    mov     EDI,params.write_addr
    mov     ES:[EDI],1010H      ; Write To Flash command
    mov     ES:[EDI],params.data
    ; Write data to 28F016SA.

WSM_not_ready1:
    mov     AX,ES:[EDI]        ; Read CSR
    test    AX,80H             ; Look at CSR.7.
    jz      short WSM_not_ready1 ; Loop while CSR.7 = 0.
    ; Poll CSR until CSR.7 = 1, indicating that WSM is ready.

    mov     BX,AX              ; Return CSR in BX.
    mov     ES:[EDI],5050H     ; Clear Status Registers command
    ret     4                  ; Return to calling routine.
compatible_byte_write endp

```

```

;=====
; PROCEDURE      compatible_suspend_to_read
; This procedure suspends an erase operation to do a read.
; It also works with a pair of 28F008SAs.
; It assumes that erase is underway.
; Param fields needed:
;      params.erase_addr: offset of 28F016SA block to erase
;      params.read_addr: offset of 28F016SA address to read
; Output: BX: CSR, duplicated in both high and low bytes
;          CX: data read from the address in params.read_addr
;=====
erase_suspend  proc  near
                mov  EDI,params.erase_addr    ; Set up offset of erase address.
                mov  ES:[EDI],BOBOH           ; Erase Suspend command

not_ready:
                mov  AX,ES:[EDI]              ; Read CSR from any address.
                test AX,80H
                jz   short not_ready
                ; Poll CSR until CSR.7 = 1, indicating that WSM is ready.

                mov  EDI,params.read_addr      ; Set up offset of read address.
                mov  ES:[EDI],FFFFH           ; Read Flash command
                mov  CX,ES:[EDI]              ; Do actual read; put result in CX.
                ; Arbitrary number of reads can be done here.
                mov  ES:[EDI],7070H           ; Read CSR command
                mov  AX,ES:[EDI]              ; Read CSR from any address.
                test AX,0040H                 ; If CSR.6 = 0, indicating that
                ; there is no erase suspended,
                jz   short continue           ; jump to continue.
                mov  EDI,params.erase_addr    ; Else set up offset of erase block.
                mov  ES:[EDI],DODOH           ; Erase Resume command

continue:
                mov  BX,AX                    ; Return CSR in BX.
                ret                            ; Return to calling routine.
erase_suspend  endp

```

```

;=====
; PROCEDURE      device_information_upload
; This procedure uploads status information into the page buffer.
; Param fields needed:
;      params.write_base: offset of 28F016SA device
; Output:  DX: Device revision number
;          AX, CX: trash
;=====
;
status_upload proc near
    mov     EDI, params.write_base
    mov     ES:[EDI], 7171H      ;Read ESR command.
    inc     EDI                  ;Move EDI up to GSR.
    inc     EDI
;
q_unavailable2:
    mov     AX, ES:[EDI]         ;Read GSR
    test    AX, 8
    jne     short q_unavailable2
    ; Poll GSR while GSR.3 = 1, indicating queue unavailable.
wsm_busy:
    mov     AX, ES:[EDI]         ;Read GSR
    test    AX, 80H
    je      short wsm_busy
    ;Poll GSR while GSR.7 = 0, indicating WSM busy.

    mov     ES:[EDI], 9999H      ;Device information Upload
                                ;command
    mov     ES:[EDI], DODOH      ;Confirmation command
    mov     ES:[EDI], 7171H      ;Read ESR command

WSM_busy2:
    mov     AX, ES:[DI]          ;READ GSR
    test    AX, 80H
    jz      short WSM_busy2
    ;Poll GSM while GSR.7 =0, indicating WSM_busy

    mov     ES:[EDI], 7272H      ;Swap Page Buffer command
    mov     ES:[EDI], 7575H      ;Read Page Buffer command
    mov     DX, [params.write_base+3] ;Put revision number in DX
    ;Revision number is 3 words above write_base in page buffer space.
    ;GSR.5 should be checked for operation success before using revision
    number.
    ret                          ;Return to calling routine.
status_upload endp

```

3

```

;=====
; PROCEDURE      ESR_block_erase
; This procedure erases a block on the 28F016SA.
; Param fields needed:
;      params.erase_addr: offset of base of 28F016SA block to erase
; Output:  BX: GSR in high byte and BSR in low byte
;          AX, DX: trash
;=====
ESR_block_erase proc      near
    mov     EDI,params.erase_addr        ; Set up offset of address.
    mov     ES:[EDI],7171H                ; Read ESR command
    inc     EDI                           ; BSR is 1 word above base of target 64K block in ESR space.
    q_full4:
    mov     AX,ES:[EDI]                   ; Read BSR
    test    AX,8
    jne     short q_full4
    ; Loop while BSR.3 of target address is 1, meaning queue full.
    mov     ES:[EDI],2020H                ; Block Erase command
    mov     ES:[EDI],D0D0H                ; Confirm command
    mov     ES:[EDI],7171H                ; Read ESR command
    ;Note that EDI still contains offset of device base + 1.
    wait_BSR74:
    mov     AX,ES:[EDI]                   ; Read BSR
    test    AX,80H
    jz      short wait_BSR74
    ;Loop while BSR.7 of target address = 0, i.e. block busy.
    mov     BL,AL                          ; Store BSR in BL.
    inc     EDI                           ; Move EDI up to read GSR.
    mov     AX,ES:[EDI]                   ; Read GSR
    mov     BH,AH                          ; Store GSR in BH.
    mov     ES:[EDI],5050H                ; Clear Status Registers command
    ret                                    ; Return to calling routine.
ESR_block_erase endp

```



```

;=====
; PROCEDURE      ESR_suspend_to_read
; This procedure suspends an erase on the 28F016SA.
; Param fields needed:
;
;      params.erase_addr: offset of base of erasing 28F016SA block
;      params.read_addr: offset of 28F016SA address to read
; Output:  BX: GSR in high byte and BSR in low byte (of erase block)
;          CX: data read from flash
;          AX, DX: trash
;=====
suspend_to_read proc      near
    mov     EDI,params.erase_addr
    inc     EDI
    ; BSR is 1 word above base of target 64K block in ESR space.
    mov     ES:[EDI],7171H      ; Read ESR command

wait_BSR75:
    mov     AX,ES:[EDI]          ; Read BSR
    test    AX,80H
    jz      short wait_BSR75
    ; Loop if BSR.7 of target address is 0, meaning block busy.

    mov     ES:[EDI],BOBOH      ; Operation Suspend command
    mov     EDI,params.read_addr ; Set up offset of read address.
    mov     ES:[EDI],7171H      ; Read ESR command
    inc     EDI
    inc     EDI
    ; GSR is 2 words above base of target 64K block in ESR space.

WSM_busy3:
    mov     AX,ES:[EDI]          ; Read GSR
    test    AX,80H
    jz      short WSM_busy3
    ; Poll GSR until GSR.7 indicates WSM is ready.

    mov     EDI,params.read_addr ; Set up offset of read address.
    mov     ES:[EDI],FFFFH      ; Write Read Flash Array command
    mov     AX,ES:[EDI]          ; Read the data

    mov     CX,AX                ; Store the result in CX.
    mov     EDI,params.erase_addr ; Set up offset of erase address.
    inc     EDI
    inc     EDI
    ; GSR is 2 words above base of target 64K block in ESR space.
    mov     AX,ES:[EDI]          ; Read GSR
    test    AX,40H
    jz      short nothing_suspended
    ; If GSR.6 indicates an operation was suspended on this device,
    ; then resume the operation.
    mov     ES:[EDI],DODOH      ; Resume command

nothing_suspended:
    mov     BH,AH                ; Store GSR in BH.
    dec     EDI                  ; Move EDI down to read BSR.
    mov     AX,ES:[EDI]          ; Read BSR
    mov     BL,AL                ; Store BSR in BL.
    ret                          ; Return to calling routine.

suspend_to_read endp

```

```

;=====
; PROCEDURE      ESR_word_write
; This procedure writes a word to the 28F016SA.
; Param fields needed:
;      params.write_base: offset of base of 28F016SA block to write
;      params.data: data word to write
;      params.write_addr: offset of 28F016SA address to write
; Output:  BX: GSR in high byte and BSR in low byte
;          AX, CX, DX: trash
;=====

```

```
ESR_word_write  proc      near
```

```

    mov  AX,7171H                ; Read ESR command
    mov  EDI,params.write_addr    ; Set up offset of write address.
    mov  ES:[EDI],AX              ; Write the command.
    mov  EDI,params.write_base    ; Get base of block to write
    inc  EDI
    ;BSR 1 word above base of target 64K block in status reg space.

```

```
q_full:
```

```

    mov  AX,ES:[EDI]              ; Read BSR
    test AX,8
    jne  short q_full
    ;Loop while BSR.3 of target address 1, meaning queue full.

```

```

    mov  AX,1010H                ; Write Byte command
    mov  EDI,params.write_addr    ; Set up offset of write address.
    mov  ES:[EDI],params.data     ; Write actual data.
    mov  ES:[EDI],7171H           ; Read ESR command
    mov  EDI,params.write_base    ; Set up offset of block base.
    inc  EDI
    ;BSR 1 word above base of target 64K block in status reg space.

```

```
wait_BSR71:
```

```

    mov  AX,ES:[EDI]              ; Read BSR
    test AX,0080H
    jz   short wait_BSR71
    ;Poll BSR while BSR.7 of target address is 0, meaning block busy

```

```

    mov  BL,AL                    ; Store BSR in BL
    inc  EDI
    ;GSR 2 words above base of target 64K block in status reg space.
    mov  AX,ES:[EDI]              ; Read GSR
    mov  BH,AH                    ; Store GSR in BH
    mov  AX,5050H                 ; Clear Status Registers command
    mov  EDI,params.write_addr    ; Set up offset of address.
    mov  ES:[EDI],AX              ; Write the command.
    ret                           ; Return to calling routine.

```

```
ESR_word_write  endp
```

```

;=====
; PROCEDURE      erase_all_unlocked_blocks
; This procedure erases all the unlocked blocks on a 28F016SA.
;      params.erase_addr: offset of base of device to erase
; Output:  BX: GSR in both high byte and low byte
;          AX, CX, DX: trash
;=====
erase_all_unlocked_blocks  proc  near

    push SI                      ; Save old SI.
    mov  EDI,params.erase_addr  ; Set up offset of address.
    mov  ES:[EDI],7171H         ; Read ESR command
    inc  EDI                    ; BSR is 1 word above base of target 64K block in ESR space.

q_full5:
    mov  AX,ES:[EDI]             ; Read BSR
    test AX,8
    jne  short q_full5
    ; Poll BSR while BSR.3 of target address is 1, meaning queue full.

    mov  ES:[EDI],A7A7H         ; Full-chip Erase command
    mov  ES:[EDI],DODOH         ; Confirm command
    mov  ES:[EDI],7171H         ; Read ESR command
    inc  EDI
    ; GSR is 2 words above base of target 64K block in ESR space.

WSM_not_ready3:
    mov  AX,ES:[EDI]             ; Read GSR
    test AX,80H
    jz   short WSM_not_ready3
    ;loop until GSR.7 indicates WSM is ready

    mov  AX,ES:[EDI]             ; Read GSR for operation success
    test AX,20H
    jz   short chip_erase_success

    ; If GSR.5 = 1, meaning that the operation was unsuccessful,
    ; go through blocks, looking for the ones which EDI didn't erase.
    xor  SI,SI                   ; Clear SI.

look_for_bad_erase:
    ; Looking at each BSR.5 for operation success.
    mov  ES:[EDI],next_block_base
    inc  EDI                     ; BSR is 1 word above base of block.
    mov  AX,ES:[EDI]             ; Read BSR
    test AX,20H
    jz   short ok_erased

; record number of bad block here

ok_erased:
    inc  SI
    cmp  SI,20H
    jl   short look_for_bad_erase
    mov  EDI,params.erase_addr  ; Set up offset of device base.
    mov  BX,ES:[EDI]             ; Read GSR for return
    mov  ES:[EDI],5050H         ; Clear Status Registers command
    pop  SI                     ; Retrieve old SI.
    ret                          ; Return to calling routine.

erase_all_unlocked_blocks endp

```

```

;=====
; PROCEDURE      lock_block
; This procedure locks a block on the 28F016SA.
; Param fields needed:
;      params.lock_addr: offset of base of 28F016SA block to lock
; Output:  BX: GSR in high byte and BSR in low byte
;      AX, DX: trash
;=====
lock_block      proc      near

                mov     EDI,params.lock_addr      ; Set up offset of address.
                mov     ES:[EDI],7171H           ; Read ESR command
                inc     EDI
                inc     EDI

q_unavailable:
                read
                test    AX,8
                jne     short q_unavailable
                ; Poll GSR while GSR.3 = 1, indicating queue unavailable.

                set_pin 56,1                      ; EDIisable write protection.
                set_pin 15,1                      ; Enable Vpp
                ; Wait for ramp if necessary.
                mov     ES:[EDI],7777H           ; Lock Block command
                mov     ES:[EDI],DODOH           ; Confirmation command
                mov     ES:[EDI],7171H           ; Read ESR command

WSM_busy:
                mov     AX,ES:[EDI]              ; Read GSR
                test    AX,80H
                jz      short WSM_busy
                ; Poll GSR while GSR.7 = 0, indicating WSM_busy.

                mov     BH,AH                    ; Store GSR
                ; Look at BSR.6 to see if block successfully locked.
                dec     EDI
                mov     AX,ES:[EDI]              ; Read BSR
                mov     BL,AL                    ; Store BSR
                mov     ES:[EDI],5050H           ; Clear Status Registers command
                ret                                ; Return to calling routine.

lock_block      endp

```



```

;=====
; PROCEDURE      status_upload
; This procedure uploads status information into the ESR from non-volatile
; status bits.
; Param fields needed:
;      params.lock_addr: offset of 28F016SA device
; Output:  BX: GSR in high byte and BSR in low byte
;          AX, CX, DX: trash
;=====
status_upload    proc    near
    mov     EDI, params.lock_addr
    mov     ES:[EDI], 7171H          ; Read ESR command.
    inc     EDI                     ; Move EDI up to GSR.
    inc     EDI
q_unavailable2:
    mov     AX, ES:[EDI]             ; Read GSR
    test    AX, 8
    jne     short q_unavailable2
    ; Poll GSR while GSR.3 = 1, indicating queue unavailable.
    mov     ES:[EDI], 9797H          ; Lock-status Upload command
    mov     ES:[EDI], DODOH          ; Confirmation command
    mov     ES:[EDI], 7171H          ; Read ESR command
WSM_busy2:
    mov     AX, ES:[EDI]             ; Read GSR
    test    AX, 80H
    jz      short WSM_busy2
    ; Poll GSR while GSR.7 = 0, indicating WSM_busy
    mov     BH, ES:[EDI]             ; Read GSR
    dec     EDI
    mov     BL, ES:[EDI]             ; Read and store BSR
    mov     ES:[EDI], 5050H          ; Clear Status Registers command
    ret
status_upload    endp

```

```

;=====
; PROCEDURE      pagebuffer_write_to_flash
; This procedure writes from page buffer to flash.
; Param fields needed:
;      params.write_base: offset of base of 28F016SA block to write
;      params.pagebuffer_word_count: number of words to write to flash
;      params.write_addr: offset of 28F016SA address to write
; Output:  BX: GSR in high byte and BSR in low byte of BX
;      AX, CX, DX: trash
;=====
pagebuffer_write_to_flash    proc    near

    push SI                    ; Save old SI.
    mov SI,params.pagebuffer_word_count    ; Use SI to count words.
; Address is where in 28F016SA flash array to begin write. The lowest
; byte of this must be identical to the start address in the page buffer.
; Low byte of byte_count must be 256 or fewer, high must be 0.
; High byte exists for future Page Buffer expandability.
    mov EDI,params.write_base    ; Offset of block base address.
    mov ES:[EDI],7171H            ; Read ESR command
    inc EDI
;BSR is 1 word above base of target 64K block in status reg space.

q_full3:
    mov AX,ES:[EDI]              ; Read BSR
    test AX,8
    jne short q_full3
    ; Loop while BSR.3 of target address is 1, meaning queue full.

    mov ES:[EDI],0COCH            ; Page Buffer Write command
    mov ES:[EDI],SI              ; Write count
    ;Only A0 valid here; low or high byte loaded depending on A0.
    mov ES:[EDI],SI
    ;A0 internally complemented; alternate byte loads; write starts.
    mov ES:[EDI],7171H            ; Read ESR command

wait_BSR73:
    mov AX,ES:[EDI]              ; Read BSR
    test AX,80H
    jz short wait_BSR73
    ;Loop while BSR.7 of target address is 0, meaning block busy.

    mov EDI,params.write_base
    inc EDI
    ;BSR is 1 word above base of target 64K block in status reg space.

    mov AX,ES:[EDI]              ; Read
    mov AX,5050H                 ; Clear Status Registers command
    mov EDI,params.write_addr    ; Set up offset of address.
    mov ES:[EDI],AX              ; Write
    pop SI                       ; Retrieve old SI.
    mov BH,ES:[EDI]              ; Read GSR
    dec EDI
    mov BL,ES:[EDI]              ; Read and store BSR
    mov ES:[EDI],5050H           ; Clear Status Registers command
    ret                          ; Return to calling routine.

pagebuffer_write_to_flash    endp

```

```

=====
; PROCEDURE sequential_pagebuffer_load
; This procedure loads the page buffer.
; Param fields needed:
;   params.write_addr: offset of origin of device
;   params.data_addr: pointer to data to be written to pg buffer
;   params.pagebuffer_word_count: number of words to write to pg
;   buffer
;   params.pagebuffer_start_addr: starting pb address of data to
;   write
; Output: AX, BX, DX: trash
=====
sequential_pagebuffer_load    proc    near

    sub    SP,2                ; Set aside room for counter.
    mov    byte ptr[BP-1],0    ; Clear high byte of counter
                                ; word.
    push    SI                 ; Save old SI.
    mov    SI,params.pagebuffer_word_count ; Put # of words to write in SI.
                                ; SP+6 must be 128 or fewer, SP+7 must be 0.
                                ; High byte exists for future Page Buffer expandability.
    mov    EDI,params.write_addr ; Set up offset of device
                                ; address.
    mov    ES:[EDI],7171H      ; Read ESR command
; Commands to control entire 28F016SA do not need to be written to any
; particular address.
    inc    EDI
    inc    EDI                ; Point EDI to GSR.

wait_for_pb2:
    mov    AX,ES:[EDI]        ; Read GSR
    test    AX,4
    jz     short wait_for_pb2 ; Poll GSR until GSR.2 indicates that a page buffer is available.

    mov    ES:[EDI],EOEOH     ; Sequential Page Buffer Load
                                ; cmd.
    ; Loads high or low byte of count register, depending on A0.
    mov    ES:[EDI],SI        ; Write
    ; Automatically loads alternate byte of count register.
    mov    ES:[EDI],SI        ; Write

; Loop through data, writing to page buffer.
    mov    byte ptr[BP-1],0    ; Load counter.
    jmp     short compare

not_done:
    mov    DX,word ptr[BP-2]   ; Put current val. of counter in
DX.
    mov    AL,params.pagebuffer_start_addr ; Get starting address in pb.
    cbw                                ; Convert it to a word.
    add    AX,DX                 ; Add to get abs. address in pb.
    cwd                                ; Convert AX to a double word.
    mov    BP+12,DX             ; Store segment of pb address
                                ; (0).

```

```

mov AX,word ptr[BP-2] ; Get current value of counter.
mov EBX,params.data_addr ; Get address of where data is.
add EBX,AX ; Add value of counter to it.
mov ES,word ptr[BX] ; Put data at that address on
; stack.
mov EDI,params.write_base ; Set up offset of address.
mov ES:[EDI],AX ; Write
inc word ptr[BP-2] ; Increment counter.

compare:
mov AX,word ptr[BP-2] ; Get current value of counter.
cmp AX,SI ; Compare to final value.
jl short not_done

; End of loop.
pop SI ; Retrieve old SI.
mov SP,BP ; Retrieve old SP.
ret ; Return to calling routine.

```

sequential_pagebuffer_load endp


```

; PROCEDURE      single_pagebuffer_load
; This procedure loads a single byte or word to a page buffer.
; Param fields needed:
;      params.write_base: offset of base of device
;      params.data: data to be written to page buffer
;      params.pagebuffer_start_addr: byte giving pb location to write
; Output:  AX: trash
;=====

```

```

single_pagebuffer_load      proc      near

      mov     EDI,params.write_base      ; Set up offset of base
address.
      mov     ES:[EDI],7171H             ; Read ESR command
      inc     EDI
      inc     EDI

wait_for_pb:
      mov     AX,ES:[EDI]                 ; Read GSR
      test    AX,4
      jz      short wait_for_pb
      ; Poll GSR until GSR.2 indicates that a page buffer is available

      mov     ES:[EDI],7474H             ; Single PB Write command
      ; Actual write to page buffer.
      add     EDI,params.pagebuffer_start_addr ; Set up offset of address.
      mov     ES:[EDI],params.data
      ; BP+4 is location in pb to write.
      ret                                  ; Return to calling routine.
single_pagebuffer_load      endp

```

```

=====
; PROCEDURE      two_byte_write
; This routine is used when BYTE# is low, i.e. the 28F016SA
; is in byte mode, to emulate a word write.
; Param fields needed: (assume existence of byte fields data_high and
; data_low)
;
;      params.write_base: offset of base of 28F016SA block to write
;      params.data_high: high data byte to write
;      params.data_low: low data byte to write
;      params.write_addr: offset of 28F016SA address to write
; Output:  BX: GSR in high byte and BSR in low byte
;          AX, CX, DX: trash
=====
two_byte_write proc      near

        mov     EDI,params.write_base      ; Set up offset of address.
        mov     ES:[EDI],7171H             ; Read ESR command
        inc     EDI
        ; BSR is 1 word above base of target 64K block in ESR space.

q_full2:
        mov     AX,ES:[EDI]                ; Read BSR
        test    AX,8
        jne     short q_full2
        ; Loop while BSR.3 of target address is 1, meaning queue full.

        mov     ES:[EDI],FBFBH             ; Two-byte write command
        ; Write low byte of data word
        mov     EDI,params.write_addr      ; Set up offset of address.
        mov     ES:[EDI],params.data_high
        mov     ES:[EDI],params.data_low

; 28F016SA automatically loads alternate byte of data register and
; initiates write. Now we check for successful completion.

        mov     ES:[EDI],7171H             ; Read ESR command
        mov     EDI,params.write_base
        inc     EDI
        ; BSR is 1 word above base of target 64K block in ESR space.

wait_BSR72:
        mov     AX,ES:[EDI]                ; Read BSR
        test    AX,80H
        jz      short wait_BSR72
        ; Poll BSR while BSR.7 of target address is 0, meaning block busy.

        mov     BH,ES:[EDI]                ; Read BSR
        inc     EDI
        mov     BL,ES:[EDI]                ; Read and store GSR
        mov     ES:[EDI],5050H             ; Clear Status Registers command
        ret                                  ; Return to calling routine.

two_byte_write endp

```

```

=====
; PROCEDURE      write_during_erase
; This procedure writes to one block while another is erasing.
; Param fields needed:
;      params.data: data word to write to 28F016SA
;      params.erase_addr: offset of 28F016SA address to erase
;      params.write_addr: offset of 28F016SA address to write
; Output:  BX: GSR in high byte and BSR in low byte
;          AX, DX: trash
=====
write_during_erase      proc      near

        mov     EDI,params.erase_addr    ; Set up offset of address.
        mov     ES:[EDI],7171H           ; Read ESR command
        inc     EDI
        ; BSR is 1 word above base of target 64K block in ESR space.
q_full6:
        mov     AX,ES:[EDI]              ; Read BSR
        test    AX,8
        jne     short q_full6
        ; Loop while BSR.3 of target address is 1, meaning queue full.

        mov     ES:[EDI],2020H           ; Write Erase Block command
        mov     ES:[EDI],D0D0H           ; Erase Confirm command
        mov     EDI,params.write_addr    ; Set up offset of address.
        mov     ES:[EDI],4040H           ; Write Word command
        mov     ES:[EDI],params.data     ; Write actual data

        ; Erase will suspend, write will take place, then erase resumes.

        mov     EDI,params.erase_addr    ; Set up offset of address.
        mov     ES:[EDI],7171H           ; Read ESR command
        inc     EDI
        ; BSR is 1 word above base of target 64K block in ESR space.
wait_BSR76:
        read
        test    AX,80H
        jz      short wait_BSR76
        ; Loop while BSR.7 of target address is 0, meaning block busy.

        mov     EDI,params.erase_addr    ; Set up offset of address.
        mov     BH,ES:[EDI]              ; Read BSR
        inc     EDI
        mov     BL,ES:[EDI]              ; Read and store GSR
        mov     ES:[EDI],5050H           ; Clear Status Registers command
        ret                               ; Return to calling routine.
write_during_erase      endp
TEXT                    ends

```

GLOSSARY OF TERMS

BSR: Block Status Register. Each BSR reflects the status of its 64 KB block.

CSR: Compatible Status Register. The CSR reflects the status of the entire device and is identical in format to the Status Register of the 28F008SA.

DSR: Device Status Register. The contains device revision number after Status Upload command.

EDI: Extended Data Index register on 80386 and higher CPUs.

ESR: Extended Status Registers. The GSR and BSRs.

GSR: Global Storage Register. The GSR provides additional information about entire device status.

RY/BY #: Output pin from the 28F016SA indicating status of current operation.

Vpp: Voltage necessary to program the 28F016SA (12V).

WSM: Write State Machine. On-board "processor" automating write, erase and other functions

ADDITIONAL INFORMATION

Order Number	Document
290489	28F016SA Datasheet
292124	AP-375 Upgrade Considerations from the 28F008SA to the 28F016SA
297372	28F016SA User's Manual
292127	AP-378 System Optimization Using the Enhanced Features of the 28F016SA
294016	ER-33 Flash Memory Technology ETOX IV

REVISION HISTORY

Number	Description
001	Original Version

APPLICATION NOTE

3

System Optimization Using the Enhanced Features of the 28F016SA

**SALIM FEDEL
PATRICK KILLELEA
FLASH APPLICATIONS ENGINEERING
INTEL CORPORATION**

October 1993

Order Number: 292127-001

System Optimization Using the Enhanced Features of the 28F016SA

CONTENTS	PAGE	CONTENTS	PAGE
1.0 INTRODUCTION	3-261	6.0 FLEXIBLE SYSTEM INTERFACE ..	3-272
1.1 Overview of Enhanced Features	3-261	6.1 Dual Chip Enables	3-272
1.2 Glossary of Terms	3-262	6.2 Dual 3.3V/5.0V Operation	3-272
2.0 INNOVATIVE ARCHITECTURAL ENHANCEMENTS	3-262	6.3 User-Selectable x16/x8 Bus Width ..	3-274
3.0 ENHANCED WRITE CAPABILITY	3-265	6.4 Open Drain RY/BY #	3-274
3.1 Page Buffer Writes to Flash	3-265	6.5 RY/BY # Configuration Modes (Level & Pulsed)	3-276
3.2 Command Queuing	3-267	7.0 CODE AND DATA PROTECTION ..	3-277
3.3 Command Prioritization	3-267	7.1 Selective Block Locking	3-277
3.4 Extended Status Registers	3-268	7.2 Master Write Protect	3-277
3.5 Automatic Erase Suspend to Write ..	3-268	7.3 Software Partitioning	3-277
3.6 Block Validity & Data Integrity	3-268	7.4 Reset Capability	3-278
3.7 Erase All Unlocked Blocks	3-269	8.0 SUMMARY	3-278
4.0 LOW POWER CONSUMPTION	3-269	LIST OF APPENDICES	3-278
4.1 3.3V Operation	3-269	APPENDIX A: COMMAND LISTINGS	3-279
4.2 Page Buffer Write Operation	3-270	APPENDIX B: REFERENCES	3-280
4.3 Automatic Power Savings	3-270	APPENDIX C: REVISION HISTORY	3-281
4.4 Deep Power-Down Mode	3-270		
4.5 Sleep	3-271		
4.6 Standby	3-271		
5.0 DENSITY IMPROVEMENT/SPACE SAVINGS	3-271		

1.0 INTRODUCTION

1.1 Overview of Enhanced Features

The Intel family of flash memory components has a new member: the 28F016SA 16-Mbit Flashfile™ memory. The 28F016SA provides superior write performance and ultra high density, making possible faster, smaller form-factor flash memory sub-systems than ever before. Flash memory cards and flash drives will reach higher levels of performance with the 28F016SA, while new applications will arise to take

advantage of special 28F016SA features, such as low power operation, data security, and flexibility of use.

This application note describes in detail the new features of the 28F016SA and the benefits of these features to the system designer and system end-user. Readers wishing technical specifications are referred to the 28F016SA data sheet; readers wishing usage guidelines are referred to the 28F016SA User's Manual.

The following table summarizes the enhanced features and advantages of the 28F016SA:

Table 1. Feature vs. Value Summary

Feature of 28F016SA	Advantage to System Designer	Advantage to End-User
FASTER WRITES, APPROACHING HARD DISK WRITE PERFORMANCE	<ul style="list-style-type: none"> • Wider range of applications • Faster installation of embedded code • Quicker code update in the field over a modem line 	<ul style="list-style-type: none"> • Faster write to flash cards, flash drives • Faster formatting of cards, drives
HIGHER DENSITY, ENABLING HIGH CAPACITY SOLID STATE MASS STORAGE	<ul style="list-style-type: none"> • New products possible • Increased data acquisition capacity • Higher capacity cards, drives, Resident Flash Arrays (RFA) • Fewer parts = better manufacturability and reliability 	<ul style="list-style-type: none"> • Smaller systems, lower weight, lower power consumption • More room for user data on flash cards & RFAs
LOW POWER CONSUMPTION	<ul style="list-style-type: none"> • Optimized system power budget 	<ul style="list-style-type: none"> • Longer battery life • Smaller batteries • Less weight
CODE/DATA PROTECTION	<ul style="list-style-type: none"> • Opportunity to ship device drivers and application code bundled with cards, RFAs and embedded systems 	<ul style="list-style-type: none"> • User is safe from accidental modification of OS or application code • Data file security
FLEXIBLE SYSTEM INTERFACE	<ul style="list-style-type: none"> • More compact systems • Optimal bus loading • More detailed status information 	<ul style="list-style-type: none"> • 28F016SA-based cards can be used to transfer data between 3.3V portable and 5.0V desktop PCs

1.2 Glossary of Terms

APS: Automatic Power Savings feature of 28F016SA.

CSR: Compatible Status Register. The CSR reflects the status of the entire device and is identical in format to the Status Register of the 28F008SA.

ESR: Extended Status Registers. The GSR and BSRs.

GSR: Global Storage Register. The GSR provides additional information about entire device status.

BSR: Block Status Register. Each BSR reflects the status of its 64 KB block.

CE₀#: Chip Enable 0. CE₀# and CE₁# both need to be asserted to activate the 28F016SA.

CUI: Command User Interface. Interface between the microprocessor and the internal memory operation.

FFS: Flash File System. Software providing file structure and maintenance for mass storage with flash.

OE#: Output Enable pin.

RFA: Resident Flash Array. Array of flash components permanently resident on a motherboard.

RFX: Resident Flash XIP.

RFD: Resident Flash Disk.

RP#: Reset Power-Down pin. Master enable switch for the 28F016SA. Formerly called PWD# on 28F008SA.

RY/BY#: Configurable Ready/Busy pin, giving status of pending operations.

WE#: Write Enable pin.

WP#: Write Protect pin. Can be used to override block lock status bit in BSRs.

WSM: Write State Machine. On-board "processor" automating Write, Erase, Lock and other functions.

XIP: Execute In Place.

2.0 INNOVATIVE ARCHITECTURAL ENHANCEMENTS

While the 28F016SA is compatible with the 28F008SA, the 28F016SA includes hardware enhancements which provide the basis for the features described in the following sections of this document.

The hardware enhancements include:

- An improved Command User Interface
- Two Page Buffers
- Queue Registers
- Extended Status Registers
- Several new pins, implementing new functions.

The Command User Interface (CUI) of the 28F016SA understands 17 new commands which simplify usage or provide access to entirely new functions. The CUI is, however, still accessed in the same manner as with the 28F008SA and all of the 28F008SA commands are valid for the 28F016SA.

Two page buffers of 256 bytes each greatly increase the effective write speed. The page buffers are written at SRAM speeds. Writes from full page buffers to the flash array can be initiated with a single command and will complete independently, freeing the CPU for other tasks.

The Queue Registers can accept up to two (2) additional commands while the current command is executing. This allows a short sequence of commands to be issued quickly, regardless of

whether the internal Write State Machine (WSM) is actually ready to process a command. The WSM will automatically fetch the next command from the queue when it is ready.

The Extended Status Registers provide more detailed information about the state of the 28F016SA, such as the state of the queue and which page buffer is selected. A Global Status Register and a set of thirty-two (32) Block Status Registers give the system designer the ability to use each 64-Kbyte block as an independent memory.

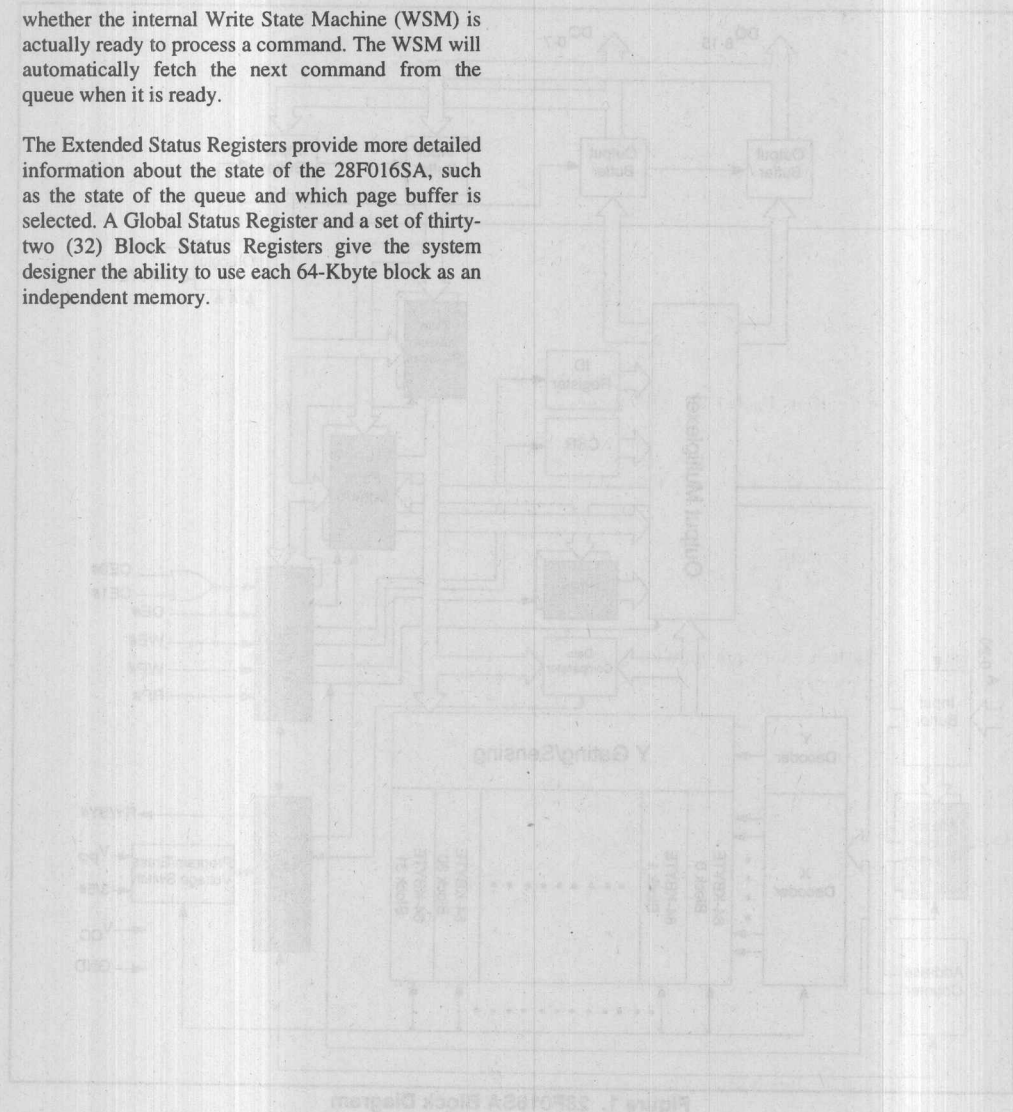


Figure 7. 28F016SA Block Diagram

ARCHITECTURAL EVOLUTION IS INDICATED WITH SHADED FUNCTIONAL BLOCKS.

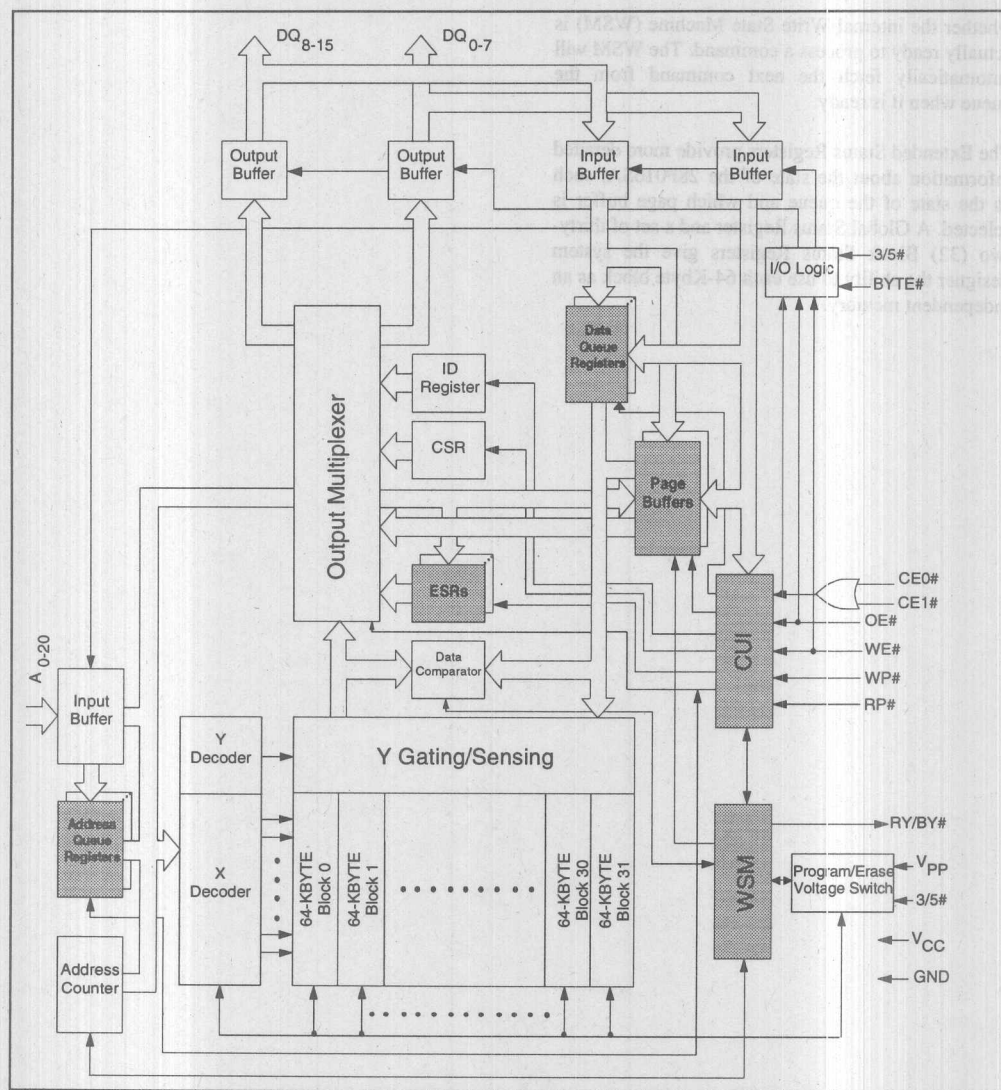


Figure 1. 28F016SA Block Diagram

ARCHITECTURAL EVOLUTION IS INDICATED WITH SHADED FUNCTIONAL BLOCKS.

3.0 ENHANCED WRITE CAPABILITY

The write performance of the 28F016SA is far superior to that of previous flash components. For a direct write to flash, the 28F016SA has a 6 μ s average word write time, a 33% improvement over the 28F008SA. Much more significant, however are the on-board page buffers and Auto Erase Suspend capability, which boost write performance much more by reducing system overhead.

3.1 Page Buffer Writes to Flash

The 28F016SA incorporates two page buffers of 256 bytes each. The page buffers can be written with SRAM-like timings and will program the flash array without any CPU overhead, providing greatly

increased write efficiencies for both short and long writes.

For example, the 28F008SA writes a byte typically in 9.2 μ s/byte, giving a device pair an effective byte write speed of 4.6 μ s since $9.2 \mu\text{s per byte}/2 \text{ devices} = 4.6 \mu\text{s per byte/device}$.

The 28F016SA, however, can write a *word* typically in 3.8 μ s from page buffer to flash memory, giving a device pair an effective byte write speed of 0.95 μ s since $3.8 \mu\text{s per word}/2 \text{ devices} = 1.9 \mu\text{s per word/device} = 0.95 \mu\text{s per byte/device}$.

This feature improves system write performance by up to 4.8 times over previous flash memory devices. When interfacing four (4) 16-Mbit devices in parallel in a 32-bit system, the sustained write speed approaches 2 MBytes/sec.

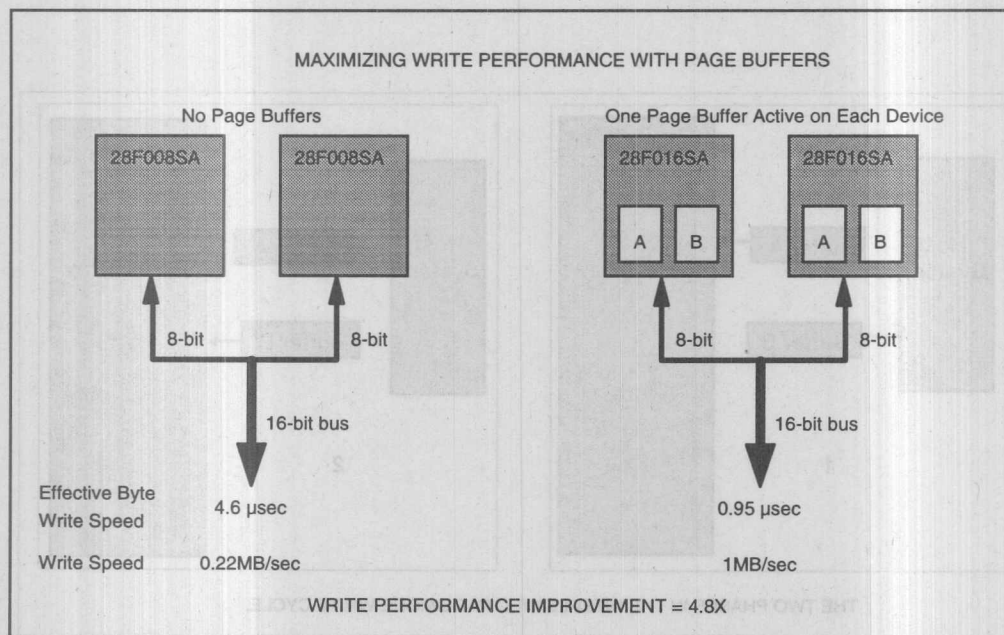


Figure 2. Page Buffer Increases Write Performance

For relatively short writes (less than 512 bytes), the CPU is free after the page buffers are loaded and the command to write to flash is issued. The 28F016SA will take care of completing the Page Buffer Write to Flash Operation. This limited form of parallel processing allows the host system to treat flash almost exactly as it would treat SRAM for short writes. For writes of less than four (4) bytes, however, it is not efficient to use the page buffers because the queue can hold three (3) simple Write commands.

For writes of more than 512 bytes, the two page buffers can be used to even greater advantage. The system can fill one buffer, issue the Write command, and then immediately begin to fill the other buffer, continuing to alternate in this way until the entire write is complete. This sort of large block writing, known as "interleaved page mode write," is efficient

because the overhead needed to set up a write from the buffers to flash is incurred only once. Also note that the write from the buffers to flash is itself faster than a write directly from the system to the flash array.

The load on the CPU is also dramatically reduced using page mode writes. While a write without the page buffer requires one setup command for every byte or word written, only 6 commands total need to be issued to set up the write of 256 bytes to a page buffer and from the page buffer to flash. While the actual write speed to the flash array is approximately 35% faster from the page buffer, there is also the benefit of huge reduction in processor overhead, with page buffer writes more than 40 times (256 commands vs. 6 commands) less of a burden on the CPU.

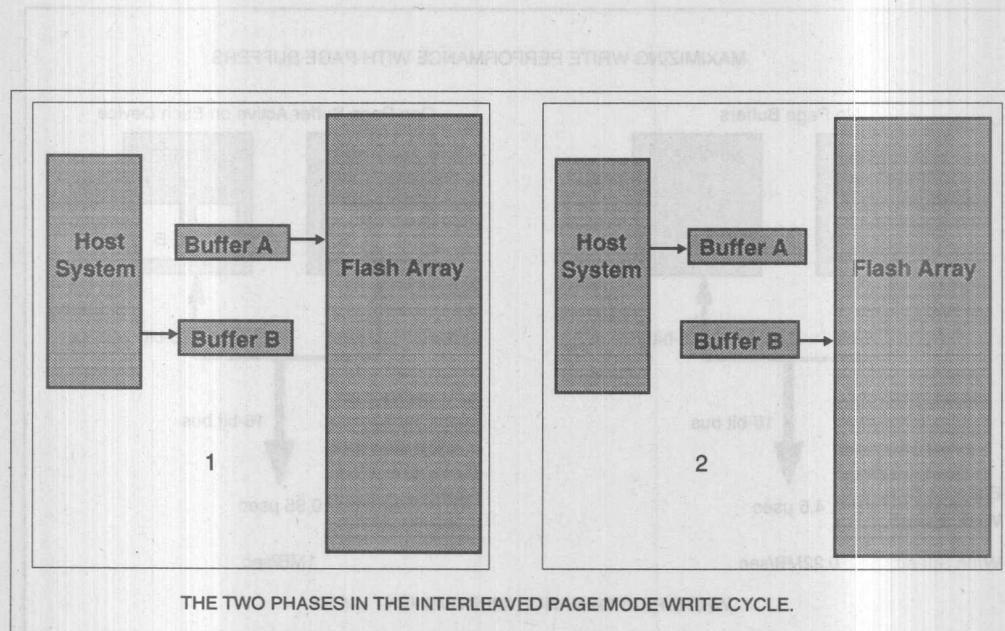


Figure 3. Interleaved Page Mode Write

3.2 Command Queuing

While the 28F008SA requires an operation to complete before the next operation could be requested, the 28F016SA allows queuing of up to two (2) additional operations while the memory executes the current operation. As a general rule, the 28F016SA has a 3-deep command queue. This eliminates system overhead when writing several bytes in a row to the array or erasing several blocks at the same time. A subset of the command-set can be queued, while the rest of the commands are executed immediately.

There is, however, an exception to the 3-deep command queue rule, which has to do with multiple Block Erase commands. If Single Block Erase commands are the only queued operations, the queue then becomes virtually 32-commands deep. This allows the user to stack many Block Erase operations very fast before a Single Block Erase

operation completes. Consult the 28F016SA User's Manual

3.3 Command Prioritization

Within the 28F016SA command queue, Write commands have higher priority than Erase commands and are executed by the WSM first, regardless of the command order. This feature helps insure that valuable data can be captured as it arrives in real time. The 28F016SA will not, however, put a write to a block ahead of an erase to the same block.

In addition, the 28F016SA prioritizes multiple Block Erase commands when queued in conjunction with Write commands. The CUI decodes the Write commands and if those commands affect a block which is in the queue for erasing, it will prioritize the Block Erase ahead of other Block Erase operations. This method allows a complete block modification to occur as fast as possible.

3

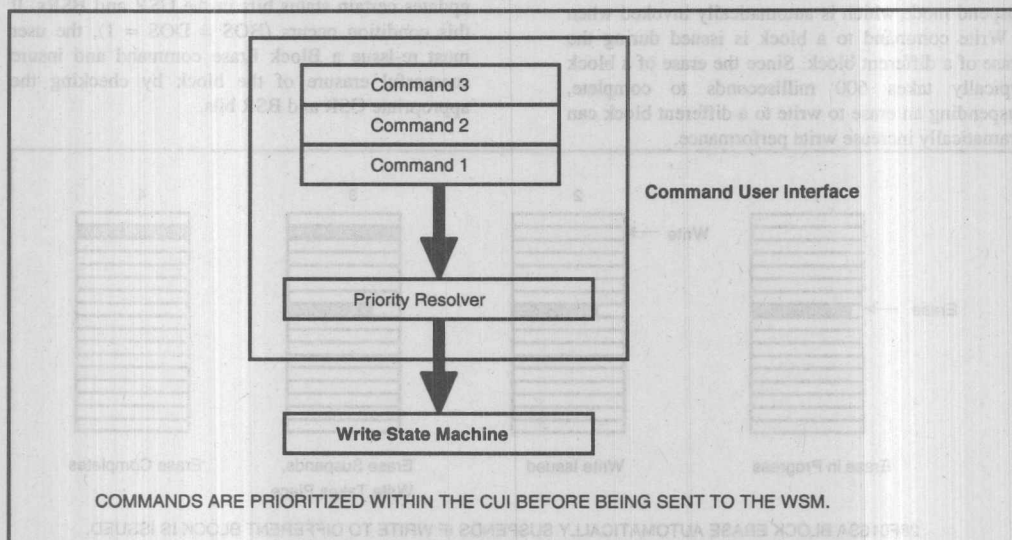


Figure 4. Command Prioritization

3.4 Extended Status Registers

The 28F016SA includes a Compatible Status Register (CSR) which is identical to the status register on the 28F008SA, a Global Status Register (GSR), which reflects the overall device status, and 32 Block Status Registers (BSRs), which are similar to the GSR except that they contain information specific to their corresponding blocks, i.e., each block has its own BSR. The value of the BSR is that it allows each block to operate essentially as an independent memory device.

Since the CPU does not have to control and monitor the details of writing a word/byte and erasing a block, it is free to perform higher priority tasks.

3.5 Automatic Erase Suspend to Write

Write performance is also enhanced by an erase suspend mode which is automatically invoked when a Write command to a block is issued during the erase of a different block. Since the erase of a block typically takes 600 milliseconds to complete, suspending an erase to write to a different block can dramatically increase write performance.

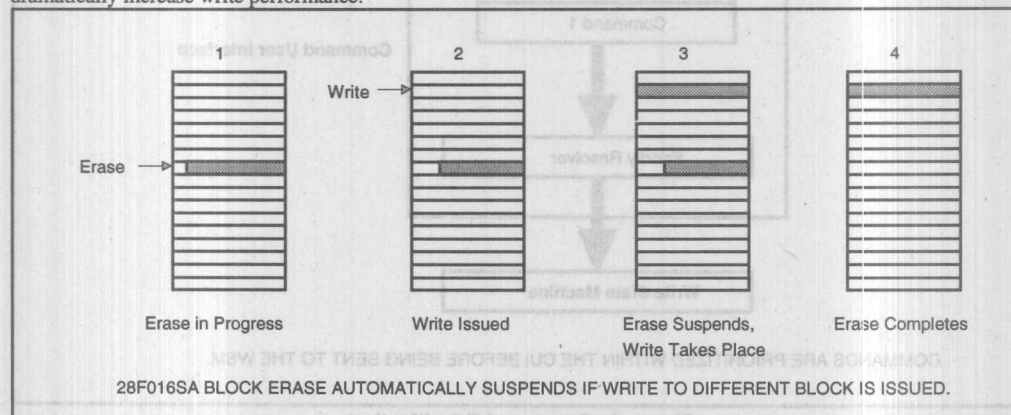


Figure 5. Automatic Erase Suspend to Write

Automatic Erase Suspend to Write is important to Microsoft's Flash Filing System (FFS) for flash memory cards. FFS needs to perform occasional background erases to maintain efficiency. These erases are much less noticeable when they can be suspended whenever the user desires access to the flash card.

3.6 Block Validity and Data Integrity

If a particular block becomes corrupted because of an interrupted Erase operation, due to an Abort command, RP# reset action or the power supply turning off, both the Block Operation Status (BOS) and Device Operations Status (DOS) bits will be set to "1," indicating an Invalid Block. This combination of status bit setting can be detected when normal operating conditions are restored and after issuing a Status Upload command, which updates certain status bits in the GSR and BSRs. If this condition occurs (BOS = DOS = 1), the user must re-issue a Block Erase command and insure successful erasure of the block by checking the appropriate GSR and BSR bits.

3.7 Erase All Unlocked Blocks

All 32 blocks of the 28F016SA can be erased using a single command, the Erase All Unlocked Blocks command. When this command and the Confirm command are issued, then all of the unlocked blocks will be erased in sequence. Locked blocks will be skipped and no error code will be returned. The BSR Block Operation Status bit can then be checked for each block to determine which block failed to properly erase and the user can re-issue single Erase commands to those particular blocks. This method improves overall system write performance in large flash memory configurations when extensive data cleanup or card formatting are required.

4.0 LOW POWER CONSUMPTION

4.1 3.3V Operation

For Read operations, the 28F016SA uses about 50% less energy in the 3.3V configuration than in the 5.0V configuration, making the 28F016SA an ideal choice for mobile computing applications as well as some power-sensitive embedded applications which use the device for infrequently updatable code storage.

For Write/Erase operations such as in Resident Flash Disk applications, the 28F016SA in 3.3V mode saves 20% / 40% energy respectively, versus the 5.0V mode. See Table 2 for detailed calculations.

Table 2. 28F016SA Typical* Power Consumption and Energy Comparison

3.3V Operation f = 4 MHz		ICC (mA)	IPP (mA)	Power (mW) ICC x 3.3V + IPP x V _{PP}	Energy (m.W.sec) Power x Time
Read Current		15	65 µA	49.7	0.41 mW.sec/Block
Write Current		8	10	146.4	32.38 mW.sec/Block
Erase Current		6	4	67.8	54.24 mW.sec/Block

5.0V Operation f = 10 MHz		ICC (mA)	IPP (mA)	Power (mW) ICC x 5.0V + IPP x V _{PP}	Energy (m.W.sec) Power x Time
Read Current		50	65 µA	250	0.82 mW.sec/Block
Write Current		25	7	209	41.09 mW.sec/Block
Erase Current		18	5	150	90 mW.sec/Block

*These numbers are based on preliminary characterization data.

Block Size= 64 KB = 32 KW

Typical word write speeds: 6 µsec (5.0V); 6.75 µsec (3.3V).

Typical block erase speeds: 600 msec (5.0V); 800 msec (3.3V)

Table 3: 3.3V to 12.0V Converters

Manufacturer	Part Number	Input (V)	Current Output	Total Components Needed	Est. Cost *
Maxim	MAX732	1.8 to 5.0	30mA	13	\$4.80
Linear Technology	1109CS8-12	2.5 to 11.0	30mA	5	\$4.00

*These cost estimates are based on published pricing at the time this Application Note was written.

NOTE:

This list is intended for example only, and in no way represents all companies that produce 12.0V conversion solutions. Since this industry develops many new solutions each year, Intel recommends that the designer contact the vendors for their latest products. Intel will continue to work with vendors to develop optimum solutions. Intel Corporation assumes no responsibility for circuitry others than circuitry embodied in Intel products.

At present, there are at least two manufacturers of 3.3V to 12.0V converters. Their solutions are described in Table 3. These solutions are given only for reference and may not be suitable for use in every system. Readers wishing additional information on DC to DC converters are referred to Intel's AP-357, "Power Supply Solutions for Flash Memory."

4.2 Page Buffer Write Operation

In addition to providing dramatically faster writes, the page buffers also save power. While the actual power saved depends on the size of the write from the page buffer, savings are typically 35% of the energy it takes to write without the page buffer, since page buffer writes are intrinsically about 35% faster while current consumption is the same.

4.3 Automatic Power Savings

Automatic Power Savings (APS) is a low power feature valid during active mode of operation. The

28F016SA incorporates "Power Reduction Control" circuitry which allows the device to put itself into a low current state when addresses are not switching (in other words, accessing the same memory location). After data is read from the memory array, the Power Reduction Control logic controls the device's power consumption by entering APS mode, where the typical ICC current is 1 mA at 5.0V and 0.8 mA at 3.3V. CPUs with a slowed clock can take advantage of this feature which is entirely automatic and transparent to the user.

4.4 Deep Power-Down Mode

The deep power-down mode is activated by the RP# pin transitioning low, which turns off all device circuitry. The only current consumed is diffusion leakage, transistor sub-threshold conduction, input leakage, and output leakage, totaling 1 μ A. However, all register contents are lost and the current operation terminates upon entering deep power-down mode.

The deep power-down feature, along with the sleep command (following section), gives the 28F016SA the ability to increase power savings dramatically by taking advantage of the fact that any one flash device is accessed only occasionally. When the device is not in use, it can be turned off so that scarce battery power is consumed only as needed. These power-saving functions can be implemented in ways entirely transparent to the end-user. The only change the user will notice is that batteries last much longer with a 28F016SA-based system.

4.5 Sleep

The Sleep command is new with the 28F016SA. Unlike deep power-down mode, during sleep mode, the status registers, page buffers, and signature ID codes can still be read. Once in sleep mode and with applied CMOS input levels, the power of the device is reduced to deep power-down current levels. The Sleep command allows the device to complete any current or pending commands before going into sleep mode. The Device Sleep Status (DSS) bit in the GSR will indicate that the device is in sleep mode. Writing the Read Array command wakes up the device out of sleep mode.

4.6 Standby

With CE₀# or CE₁# high, the memory will be in standby mode. This mode turns off much of the device's circuitry and reduces device power consumption. The outputs are in a high-impedance state independent of the state of OE# pin. If the WSM is executing a command when the device is disabled, the operation is allowed to continue. During this time the power consumption remains at the non-standby level until the operation completes. The output buffers and most of the input buffers on the chip are disabled during standby mode.

5.0 DENSITY IMPROVEMENT/ SPACE SAVINGS

The 28F016SA is twice as dense as the 28F008SA, allowing smaller systems, lower weight, and lower power consumption than ever before—crucial selling points in the highly competitive mobile PC market.

Flash densities are now on a par with DRAM densities, an important step toward the use of flash as non-volatile executable memory (Resident Flash Array), which provides "instant on" boot capability and instant access to applications and data stored in flash. RFAs also reduce the amount of necessary system DRAM.

In the removable storage market, the density of the 28F016SA will make conversion to flash the most attractive option when considered along with the properties which set flash apart from other storage media, such as non-volatility, low power consumption, and ruggedness. Flash-based data and code storage media such as PCMCIA memory cards are already on the shelf, along with PCMCIA-ATA flash drives. PCMCIA cards in particular will open new distribution channels for software since cards based on the 28F016SA now have sufficient capacity for most large commercial programs. While cost/megabyte is not yet competitive with magnetic media, the XIP or "eXecute In Place" ability of flash cards provides software distributors and end-users with a compelling reason to consider flash cards.

The density of the 28F016SA is also driving entirely new applications, such as solid-state digital photography and audio recording, which require memory capacities which were not previously economical in flash.

The point to remember is that two (2) megabytes of 70 ns randomly accessible code or data can now be stored in a rewritable nonvolatile medium of less than three (3) square centimeters—a 30% improvement over the 28F008SA.

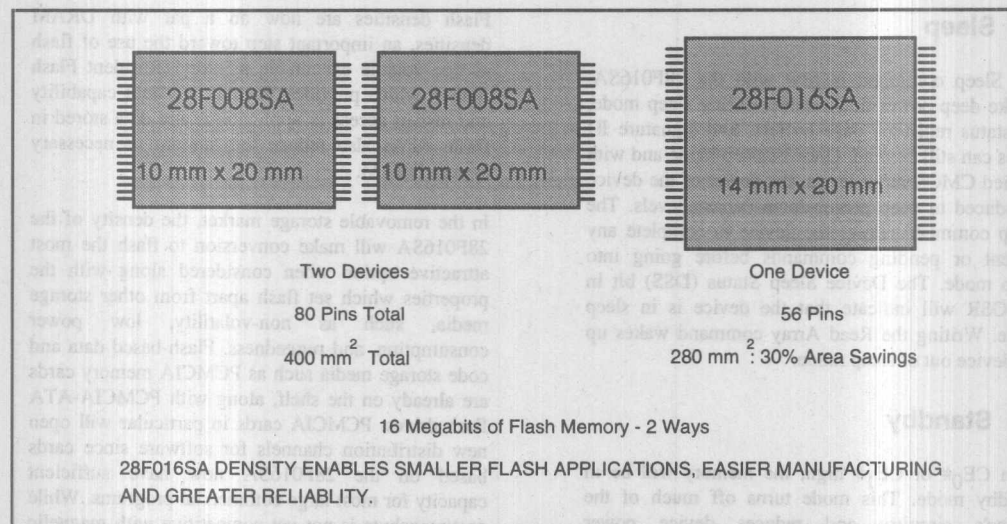


Figure 6. 28F016SA vs. 28F008SA Area Comparison

6.0 FLEXIBLE SYSTEM INTERFACE

6.1 Dual Chip Enables

The 28F016SA implements a dual chip-enable function with two input pins, $CE_0\#$ and $CE_1\#$, which together have exactly the same functionality as the regular chip-enable pin on the 28F008SA. The 28F016SA uses the logical combination of these two signals to enable or disable the entire chip. Both $CE_0\#$ and $CE_1\#$ must be active to enable the device. If either one becomes inactive, the chip will be disabled. This feature allows the system designer to reduce the number of decoding pins used in a large array of 16-Mbit devices. For square arrays, it can be seen that the number of lines needed to control $n \times n$ chips is 2 times n . For example, in a square array of sixteen 28F016SAs, only 8 lines are needed instead

of 16 (see Figure 7). For larger memory arrays, the reduction in decoding signals increases significantly.

6.2 Dual 3.3V/5.0V Operation

The portable PC market demands that components be able to operate at 3.3V. On the other hand, most desktop systems operate at 5.0V. The 28F016SA resolves this conflict with a dual operating voltage capability. A $3/5\#$ input pin makes it possible to use the 28F016SA in both 3.3V and 5.0V systems interchangeably. The $3/5\#$ signal pin from the system informs the device about the supply voltage being used. This information is used by the 28F016SA to optimize itself for the input supply voltage. Data written using one supply voltage will always be valid using the other supply voltage. A 28F016SA-based flash memory card is thus able to transfer data from a 3.3V notebook or handheld PC to a 5.0V desktop PC.

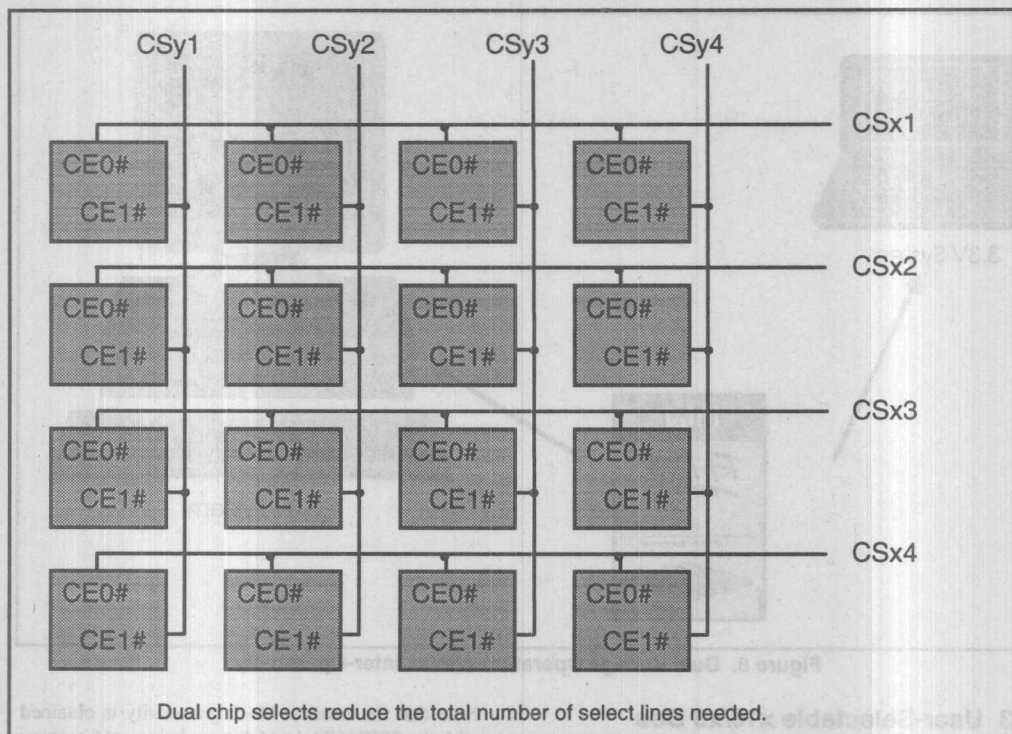


Figure 7. Dual Chip Selects in a Bank Configuration

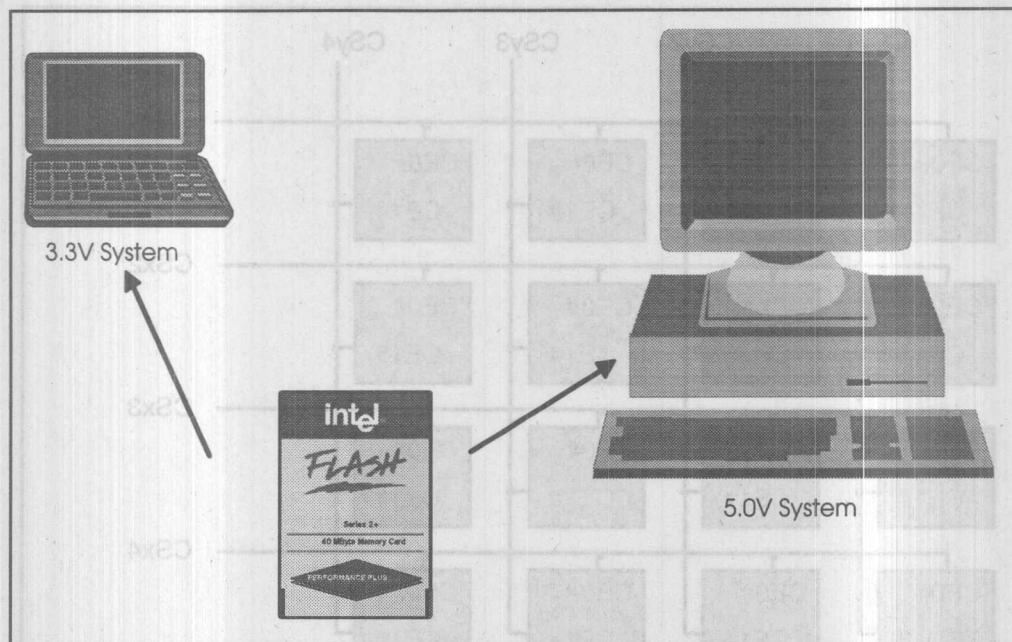


Figure 8. Dual Voltage Operation Allows Inter-Operability

6.3 User-Selectable x16/x8 Bus Width

While the 28F008SA's interface to the system bus is strictly x8, the 28F016SA's BYTE# pin allows either a x8 or x16 bus interface. The system designer now has a choice between three (3) different configurations in both 16-Bit and 32-Bit systems, allowing optimization of the effective block granularity, the space required, and the minimum memory configuration. See Table 4 for details.

Note that the smallest block granularity is obtained with the 28F016SA in 16-bit mode in a x16 system. The most efficient and smallest memory configuration is obtained with the 28F016SA in x16 mode in a 32-bit system.

6.4 Open Drain RY/BY#

The RY/BY# pin is an open drain output pin to allow the designer to Wire-OR multiple RY/BY# pins in a large memory array, saving on the number of control pins which are dedicated to this function.

Table 4. Configuration Options

System Size	Parameter	28F008SA (x8 only)	28F016SA, x8	28F016SA, x16
16-Bit System	Effective Block Size	128 KB	128 KB	64 KB
	Minimum Configuration	Two Devices: 2 MB	Two devices: 4 MB	One device: 2 MB
32-Bit System	Effective Block Size	256 KB	256 KB	128 KB
	Minimum Configuration	Four Devices: 4MB	Four devices: 8 MB	Two devices: 4 MB

3

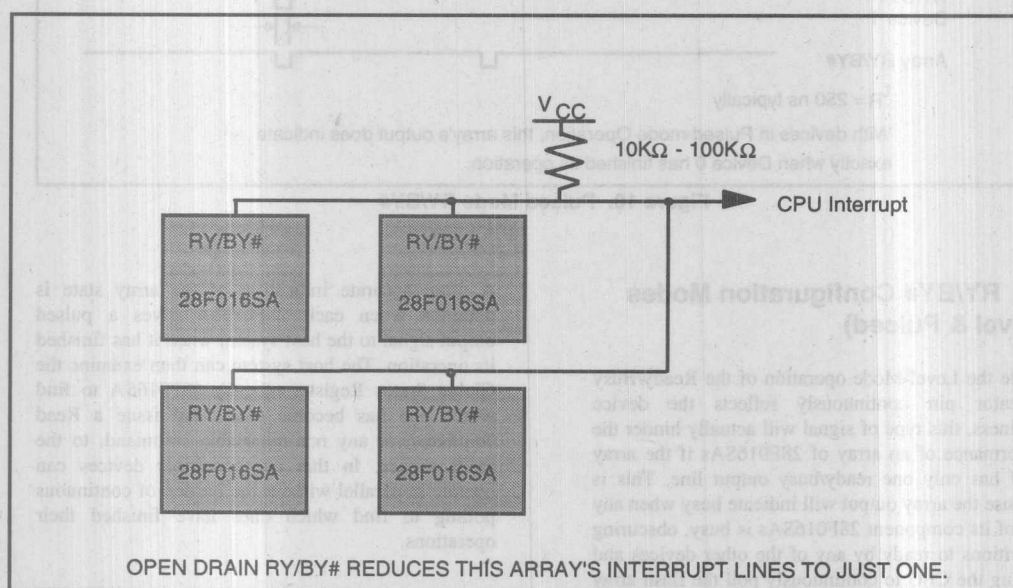


Figure 9. Open Drain RY/BY#

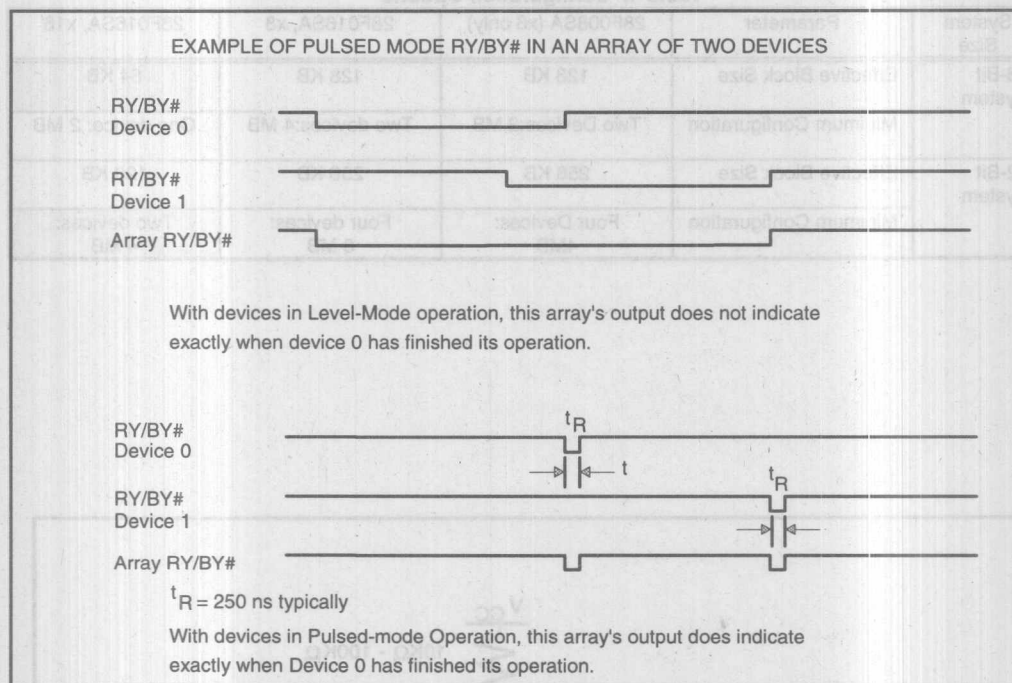


Figure 10. Pulsed Mode RY/BY#

6.5 RY/BY# Configuration Modes (Level & Pulsed)

While the Level-Mode operation of the Ready/Busy indicator pin continuously reflects the device readiness, this type of signal will actually hinder the performance of an array of 28F016SAs if the array itself has only one ready/busy output line. This is because the array output will indicate busy when any one of its component 28F016SAs is busy, obscuring transitions to ready by any of the other devices and forcing the CPU to continuously poll the flash array to find ready devices.

A more accurate indication of the array state is obtained when each 28F016SA gives a pulsed output signal to the host system when it has finished its operation. The host system can then examine the Global Status Register of each 28F016SA to find which one has become ready and issue a Read command, or any non-queueable command, to the ready device. In this way, multiple devices can operate in parallel without the burden of continuous polling to find which ones have finished their operations.

The 28F016SA incorporates a RY/BY# pin which can be configured four ways:

- Level Mode (Default)
- Pulse-on-Write Only
- Pulse-on-Erase Only
- Disable

Level mode is the default mode. In this configuration, the state of the WSM is continuously indicated by the RY/BY# pin, which is an open drain output pin pulled high through an external pull-up resistor when the WSM is ready. This feature allows the user to OR-tie RY/BY# pins of multiple devices together in flash memory arrays such as Resident Flash Array or flash drive applications, saving control logic and simplifying board design.

Pulse-on-write mode will cause RY/BY# to pulse low at the completion of Page Buffer Write to Flash operations only. This is useful for controlling interleaved page mode writes.

Pulse-on-erase mode will cause RY/BY# to pulse low at the completion of Block Erase operations, including at the end of each Block Erase during an Erase All Unlocked Blocks operation.

The RY/BY# pin can also be disabled so that it will always report a READY condition. Disabling the RY/BY# pin has no impact on the status registers.

7.0 CODE AND DATA PROTECTION

7.1 Selective Block Locking

While the 28F008SA provides data security through the RP# pin (formerly PWD#) and the intrinsic non-volatility of flash, it does not have the ability to provide selective locking of some blocks while leaving others available for writing and erasing. The 28F016SA, however, provides the ability to selectively lock any 64-Kbyte block to protect critical code or data. Each block on the 28F016SA

has an associated non-volatile lock-bit which determines the lock status of that block.

7.2 Master Write Protect

A WP# (Write Protect) pin activates the block lock-bits, preventing any Write or Erase of blocks which have their lock-bits set. When the WP# pin is asserted (low) and a block's lock-bit is set, the user is safe from accidentally damaging or modifying the data in that block.

7.3 Software Partitioning

The greatest benefit of the 28F016SA's block locking feature is that it is possible for OEMs and software developers to bundle applications or operating system code in flash memory cards or in RFAs, providing an entirely new medium for software distribution. Software distributed in this way is safe from accidental user overwrites, and yet capable of in-system updates to accommodate new versions. When 28F016SA-bundled code or data needs to be updated, raising WP# high provides a temporary override of the block locking mechanism so that locked blocks on the device may be written to.

The advantage of partitioning a 28F016SA-based memory card into locked and unlocked sections is that a user can keep an application and the files created with that application together. For example, a spreadsheet program and all of the spreadsheets a user has created with that program can be stored on one flash card. Such an arrangement gives the user a new kind of portability, one which allows him or her to carry all of the work in his or her pocket with instant access to the application and files anywhere a compatible PC is available. Hence, the data locking features of the 28F016SA complements its ability to act as executable system memory, providing the end-user with a solution which provides portability, safety, convenience, low power and high speed of access which no other medium can claim.

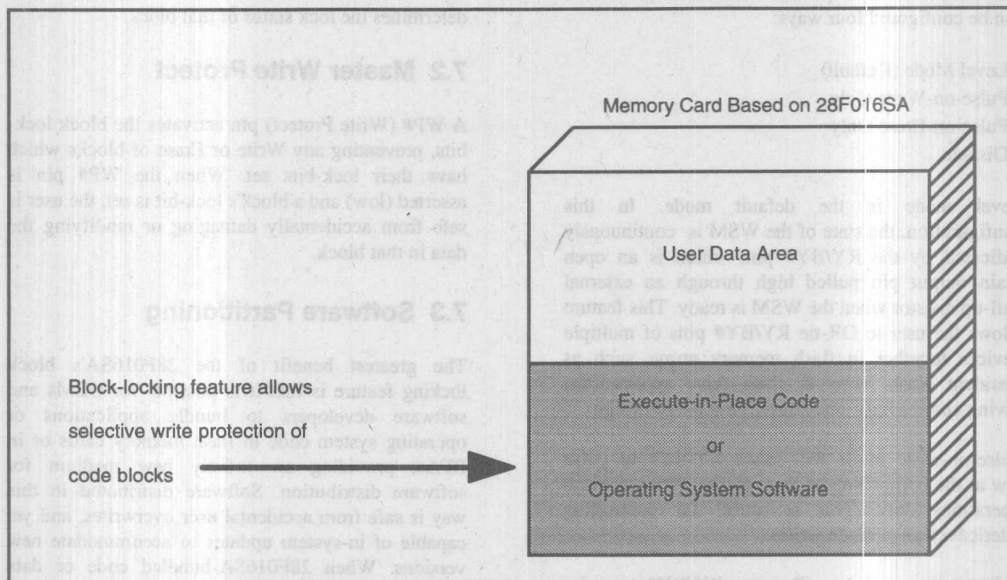


Figure 11. Block Locking and Code/Data Partitioning

7.4 Reset Capability

The 28F016SA provides complete protection of flash contents through the Reset/Power-down (RP#) pin. RP# locks the flash array from spurious writes and places the outputs in a high impedance state. If asserted during write/erase modes, RP# low aborts the current operation in progress, cancels all pending WSM commands, flushes the command queue and clears the status registers.

RP# is used both as a power conservation feature and as a data protection feature. An example of when RP# is useful for data protection is during power-up, when other inputs to the 28F016SA may be in an indeterminate state. Holding RP# low until the power supplies reach operating levels and all input signals become stable, guarantees maximum protection for the device. The use of RP# for power conservation is discussed in section 4.4. The reader

wishing more detail on the use of the RP# pin is referred to the 28F016SA User's Manual.

8.0 Summary

This application note discusses the key features and benefits of the revolutionary 16-Mbit device architecture and their impact on system and software designs. End-user benefits from these enhanced features are brought to light with respect to the wide range of new applications enabled by the 28F016SA chip and 28F016SA-based system products.

LIST OF APPENDICES

- A: Command Listings
- B: References
- C: Revision History

APPENDIX A 28F016SA COMMAND LISTINGS

	Command Codes (Hex)	Device Mode
28F008SA- Compatible Commands	00H	Invalid/Reserved
	10H	Word/Byte Write
	20H	Single Block Erase
	40H	Word/Byte Write
	50H	Clear Status Registers
	70H	Read CSR
	90H	Read ID Codes
	B0H	Erase Suspend
	D0H	Confirm/Resume
	FFH	Read Flash Array
28F016SA Performance- Enhancement Commands	0CH	Page Buffer Write to Flash
	71H	Read GSR or BSRs
	72H	Page Buffer Swap
	74H	Single Load to Page Buffer
	75H	Read Page Buffer
	77H	Lock Block
	80H	Abort
	96H	RY/BY# Reconfigurations
	01H	RY/BY# Enable to Level Mode
	02H	Pulse-On-Write
	03H	Pulse-On Erase
	04H	RY/BY# Disable
	97H	Upload Status Bits
	99H	Upload Device Information
	A7H	Erase All Unlocked Blocks
	E0H	Sequential Load to Page Buffer
	F0H	Sleep
	FBH	Two-Byte Write

APPENDIX B REFERENCES

DOCUMENT	ORDER NUMBER
28F016SA 16 Mbit (1 Mbit x 16, 2 Mbit x 8) FlashFile™ Memory Data Sheet.....	290489
28F016SA 16 Mbit FlashFile™ Memory User's Manual	297372
AP-357 Power Supply Solutions for Flash Memory	292092
AP-359 28F016SA Hardware Interfacing	292094
AP-360 28F008SA Software Drivers	292095
AP-375 Upgrade Considerations from the 28F008SA to the 28F016SA.....	292124
AP-377 The 28F016SA Software Drivers.....	292126
ER-33 ETOX™IV Flash Memory Technology	294016

APPENDIX C REVISION HISTORY

Number	Description
1.0	Original Version

ENGINEERING REPORT

APPENDIX C
REVISION HISTORY

Number	Description
1.0	Original Version

The Intel 28F008SA Flash Memory

8

ALAN BUCHECKER
JERRY KREIFELS
MEMORY COMPONENTS DIVISION

October 1993

CONTENTS	PAGE
INTRODUCTION	3-284
TECHNOLOGY OVERVIEW	3-284
DEVICE ARCHITECTURE	3-285
Array Organization	3-285
Write/Erase Automation	3-285
Command User Interface (CUI)	3-286
Write State Machine (WSM)	3-286
Status Register	3-287
Ready/Busy Indication (RY/BY#)	3-287
Internal Oscillator	3-287
Supply Voltage Sensing	3-287
Reset/Power-Down	3-288
Block Erase	3-289
Erase Suspend/Resume	3-290
Byte Write	3-290

Byte write is accomplished with the standard EPROM mechanism of channel-by-channel injection from the cell drain junction to the floating gate. During this process, the control gate and the cell drain are held at high voltage and the internal WSM regulates the internal byte-write algorithm, including margin verification, to set the correct number of pulses to write and delete. Byte write typically requires 100 μ s.

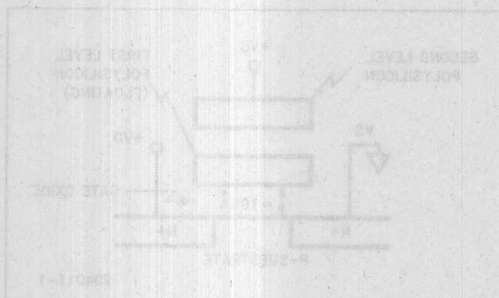


Figure 1. ET0X Flash Memory Cell

CONTENTS	PAGE
DEVICE CHARACTERIZATION	3-291
AC and DC Parameters	3-291
Energy/Power Consumption	3-291
Byte-Write/Block-Erase Times	3-291
DEVICE RELIABILITY	3-291
Byte-Write/Block-Erase Cycling	3-291
Data Protection	3-292
SUMMARY	3-292
OTHER REFERENCES	3-292
SUPPLEMENTARY INFORMATION	3-293

The ET0X is a non-volatile memory device that is designed for high-performance applications. It is a 1T1C1F1M1 cell structure, which is a combination of a 1T1C1F1M1 cell and a 1T1C1F1M1 cell. The ET0X is designed for high-performance applications and is available in a variety of packages.

The ET0X is a non-volatile memory device that is designed for high-performance applications. It is a 1T1C1F1M1 cell structure, which is a combination of a 1T1C1F1M1 cell and a 1T1C1F1M1 cell. The ET0X is designed for high-performance applications and is available in a variety of packages.

The ET0X is a non-volatile memory device that is designed for high-performance applications. It is a 1T1C1F1M1 cell structure, which is a combination of a 1T1C1F1M1 cell and a 1T1C1F1M1 cell. The ET0X is designed for high-performance applications and is available in a variety of packages.

The ET0X is a non-volatile memory device that is designed for high-performance applications. It is a 1T1C1F1M1 cell structure, which is a combination of a 1T1C1F1M1 cell and a 1T1C1F1M1 cell. The ET0X is designed for high-performance applications and is available in a variety of packages.

INTRODUCTION

The ETOX III (EPROM tunnel oxide) 28F008SA is a high-density product offering from Intel's second generation of flash memory devices. This 1,048,576 x 8 memory with its symmetrical blocking (16 blocks x 64 Kbytes), very high cycling endurance, on-chip write/erase automation, and erase-suspend/resume capability can be termed a block-alterable non-volatile RAM. In addition to selective block erasure, integrated Command User Interface (CUI), Write State Machine (WSM), Status Register, and deep power-down capability, the 28F008SA adds a dedicated READY/BUSY output (RY/BY#). This new feature provides immediate hardware signaling of byte-write/block-erase completion and erase-suspend/resume actuation.

Flash memories combine inherent non-volatility with in-system alterability of device contents. Advances in process control have allowed development of a double-polysilicon single-transistor flash memory capable of 100,000 write/erase cycles per block. The 28F008SA electrically erases all bits in a block via electron tunneling. The EPROM programming mechanism of hot-electron injection is employed for high-performance electrical byte write as required for file and data storage applications.

The Command User Interface and Status Register interface to power-up/down protection, address/data latches, and the Write State Machine (which in turn controls internal byte write, block erase, cell-margin circuits, and the dedicated READY/BUSY status output). These features augment prior flash memory circuitry to optimize Intel's 28F008SA for microprocessor-controlled byte write and block erase.

Read timing parameters are comparable to those of CMOS DRAMs, SRAMs, EPROMs, and EEPROMs. The 85 ns access time results from a memory cell-current of approximately 70 μ A, low-resistance polysilicide wordlines strapped with metal, advanced scaled periphery transistors, and an optimized data-out buffer.

The dense one-transistor cell structure, coupled with high array efficiency, yields a one-megabyte die measuring 539 by 286 mils.

TECHNOLOGY OVERVIEW

Intel's ETOX III flash memory technology incorporates advances from ETOX I and ETOX II processes and leverages over two decades of EPROM manufacturing experience. Using advanced 0.8 μ m double-polysilicon N-well/P-well CMOS technology, the 1,048,576 x 8-bit flash memory employs a 2.5 μ m x 2.9 μ m single-transistor cell affording array density equivalent to comparable EPROM technology, and twice that of Intel's ETOX II process. The ETOX III flash memory cell is identical to EPROM, with an additional source implant which optimizes erase performance. Figure 1 shows a cross-section of the flash memory cell.

High-quality tunnel oxide under the single floating polysilicon gate promotes electrical erasure. All cells within the selected block are simultaneously erased via Fowler-Nordheim tunneling. Applying 12V to block source junctions and grounding the control gates erases all cells within that block. The internal WSM controls the automated block-erase algorithm, including pre-erase conditioning (i.e., pre-programming all block bits) and margin verification, in response to user requests relayed by the CUI. WSM-controlled block erasure, including pre-programming, typically requires 1.6 seconds.

Byte write is accomplished with the standard EPROM mechanism of channel hot-electron injection from the cell drain junction to the floating gate. Bringing both the control gate and the cell drain to high voltage initiates programming. The WSM regulates the internal byte-write algorithm, including margin verification, after the correct command sequence is written and decoded. Byte write typically requires 9 μ s.

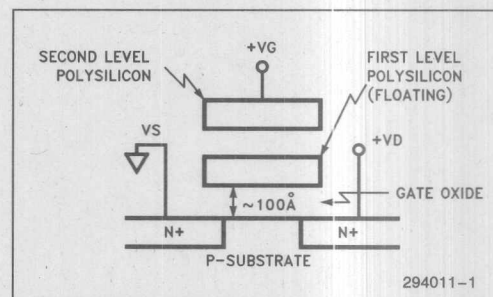


Figure 1. ETOX Flash Memory Cell

DEVICE ARCHITECTURE

Array Organization

The 28F008SA is a 1,048,576 x 8 memory comprised of 2048 rows by 8192 columns. Array layout is segmented as four quadrants, each 1024 rows by 2048 columns. Access time is reduced by limiting column length to 1024 cells. The polysilicon row is strapped in metal every 512 columns to reduce wordline delay. Two row decoders run vertically between quadrants, and column decoders run horizontally between quadrants. Figure 2 shows block placement and array organization. A die photo of the chip is shown in Figure 30.

Each quadrant is subdivided into four 64-Kbyte blocks. Each block source is electrically isolated from the source of other blocks. This allows individual block erase without altering data in the remaining 15 blocks.

Each block is further subdivided into eight Input/Outputs. Data for I/O₀ is stored in the left-most 64 columns, with the next 64 storing data for I/O₁, etc.

Rows in the upper quadrants are numbered 0–1023 from top to bottom; lower quadrant rows similarly 1024–2047.

Addresses A₉–A₀ select one of 1024 rows, while A₁₉ selects upper or lower decoder. Row address lines are decoded sequentially for selection. Row address bitmaps are listed in Table 3.

Columns are numbered 0–8191 from left to right, top to bottom. Addresses A₁₉–A₁₆ select one of 16 blocks, while A₁₅–A₁₀ select eight of the 512 columns within that block. These ten address lines are also decoded sequentially to access all 8192 columns. Block address bitmaps are listed in Table 4; column address bitmaps are listed in Table 5.

Write/Erase Automation

Intel's 28F008SA contains an on-chip Command User Interface, Write State Machine, Status Register, and address/data latches to dramatically simplify user interface. This combination of functional units reduces microprocessor control complexity of byte-write, block-erase, erase-suspend/resume, Status Register read/clear, ID read, and array read operations. Figure 3 shows the 28F008SA block diagram.

3

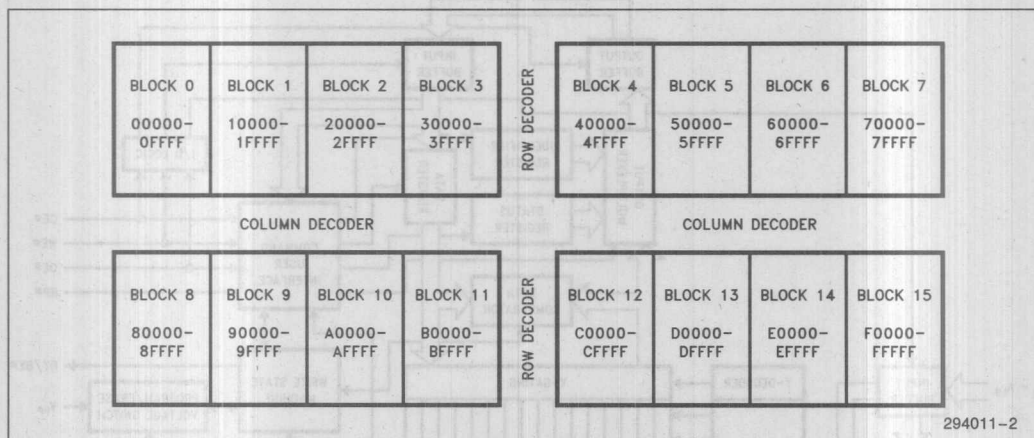


Figure 2. 28F008SA Block Placement and Array Organization

Status Register

The internal Status Register contains a full complement of activity status bits. These bits and their meaning (1,0) are:

- SR.7: WSM status (READY/BUSY)
- SR.6: Erase-suspend status (ERASE SUSPENDED/
ERASE IN PROGRESS OR COMPLETED)
- SR.5: Block-erase status (ERROR/SUCCESS)
- SR.4: Byte-write status (ERROR/SUCCESS)
- SR.3: V_{pp} status (LOW/OK)

All bits are set by the WSM, and read via the CUI. The WSM can only set SR.3, SR.4, and SR.5; it cannot clear them. They remain set until the CUI processes a clear Status Register command. There are two reasons for operating in this fashion.

First is synchronization; the WSM does not know when the host CPU has read the Status Register, therefore does not know when to clear it.

Secondly, allowing system software to control reset adds flexibility to the way this device may be used. The CPU may write several bytes or erase several blocks back-to-back while monitoring RY/BY# or polling SR.7 to determine when the next byte-write or block-erase command can be given. When all bytes are written, or all blocks erased, the system polls the other status flags to determine if all operations were successful or if an error occurred. While other approaches require the controlling microprocessor to watch for non-completion of write or erase within a specified time to indicate an error, this implementation requires no external system timers or software timing loops. As such, the system can reduce its polling overhead while still identifying any potential error conditions.

Status Register contents are driven to device outputs on the falling edge of CE# or Output Enable (OE#), whichever occurs last in the read cycle. CE# or OE# must be toggled to update Status Register contents.

Ready/Busy Indication (RY/BY#)

A dedicated output pin, RY/BY#, provides additional indication of WSM activity. This capability allows both hardware signal of status and/or software polling to determine activation, completion, or suspension of internal byte-write/block-erase operations. Hardware signaling minimizes both CPU overhead and system power consumption.

Internal Oscillator

The WSM is designed using clocked logic circuits. An on-chip ring oscillator generates the clock signals. The frequency of a standard ring oscillator varies with processing, temperature and supply voltage. An improved design, used on the 28F001BX and 28F008SA, minimizes these variations.

The switching current of each stage in the ring oscillator is controlled by a current reference which varies linearly with V_{CC}. The trip point of each ring oscillator inverter also varies linearly with V_{CC}. These two effects offset each other, and the resulting oscillator period is proportional to RC with only a small dependence on V_{CC}.

An on-chip resistor sets the value of R. The gate capacitance of the inverters in the ring oscillator sets the value of C. Process variations in these values are reduced by trimming the period of each oscillator during manufacturing. The resistor is the only source of temperature variation.

Supply Voltage Sensing

The LOWV_{CC} and LOWV_{pp} generation circuit is shown in Figure 4. Power supply voltages (V_{CC} and V_{pp}) are divided down and compared to a reference voltage. If V_{REF} is greater than the divided power supply voltage, the LOWV_{CC} or LOWV_{pp} signal is driven high. The generated V_{REF} level is supply-voltage independent to the first order.

Positive power to the circuit is supplied by M1 and M2. M1 and M2 sources are pulled up to the higher of (V_{pp} - V_{tn}) or (V_{CC} - V_{tw}). V_{tn} is the threshold of an implanted N-channel device, about 0.9V. V_{tw} is the threshold of a native N-channel device, about 0V. This scheme ensures that the circuit works regardless of the applied supply voltages.

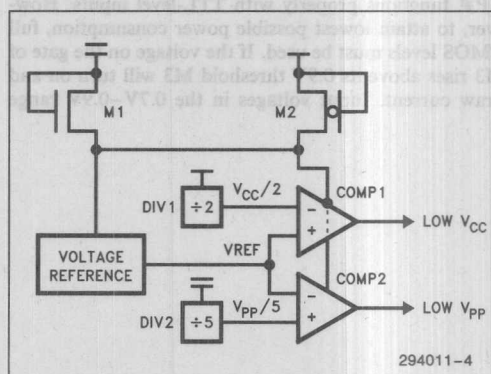


Figure 4. Low Power Detector Circuit

The $LOWV_{CC}$ signal is used by the byte-write and block-erase circuits, as well as the CUI and WSM. If $LOWV_{CC}$ is active, the CUI will not accept user writes and resets to an array read condition. The WSM is similarly reset by $LOWV_{CC}$. The $LOWV_{pp}$ signal is used by the WSM; if V_{pp} drops below the high-voltage detector trip point during byte write or block erase, the Status Register's low V_{pp} bit is set and WSM operation halts. The system must clear the Status Register before any subsequent byte-write or block-erase operations can succeed.

Reset/Power-Down

The 28F008SA incorporates a deep power-down mode that reduces I_{CC} and I_{pp} to typically $0.20 \mu A$ and $0.10 \mu A$ respectively. $RP\#$ low selects deep power-down mode. When $RP\#$ is high, the device can be placed in an active or standby mode depending on $CE\#$'s state.

Deep power-down is similar to standby except that all circuits excluding the $RP\#$ buffer are turned off. This mode greatly reduces power consumption, but requires more time to transition the device into an active mode. A read wake-up time (t_{PHQV}) is required from $RP\#$ switching high until output and sense circuitry become fully functional and data can be read from the part. Similarly, a write wake-up time (t_{PHWL}) is needed before the CUI recognizes writes. After this interval, normal operation is restored; the CUI is reset to read-array mode and the Status Register is cleared to 80H.

A diagram of the power-down circuit is shown in Figure 5. The TTL buffer formed by M1–M3 disables the low-power detect circuits, the redundancy-address flash bits, and the $CE\#$ TTL buffer formed by M4–M6. In previous Intel flash devices, these circuits were always enabled. Turn-on delays of these circuits determine $RP\#$ access time and write specifications.

$RP\#$ functions properly with TTL-level inputs. However, to attain lowest possible power consumption, full CMOS levels must be used. If the voltage on the gate of M3 rises above its $0.9V$ threshold M3 will turn on and draw current. Input voltages in the $0.7V$ – $0.9V$ range

could cause enough subthreshold conduction in M3 to exceed the I_{CC} deep power-down current (I_{CCD}) specification. This is why $RP\#$'s input voltage is specified as $GND \pm 0.2V$.

$RP\#$ also functions as a hardware reset to the WSM and CUI. If $RP\#$ is driven active during byte-write, block-erase, or erase-suspend operation, that operation is aborted leaving the addressed memory locations in an unknown state. The Status Register is cleared, and CUI is set to array read. The aborted operation (byte write or block erase) must be repeated with $RP\#$ inactive to obtain a valid condition in the memory array.

Reset using $RP\#$ should be restricted to system reset only (as in the case of power supply failure), and should not be used as a software means to terminate byte-write or block-erase operations.

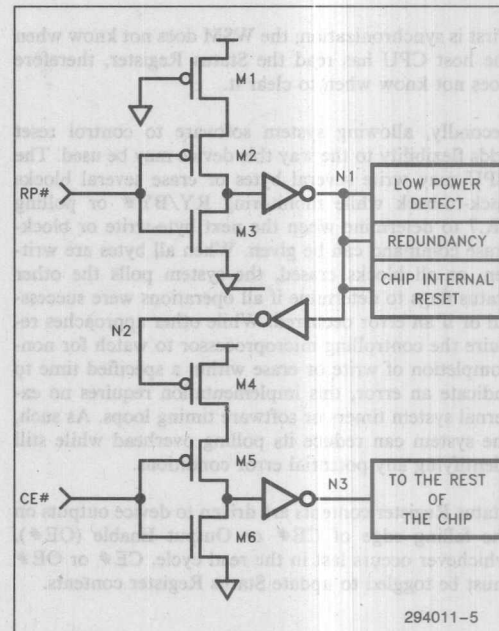


Figure 5. Power-Down and Reset Functions

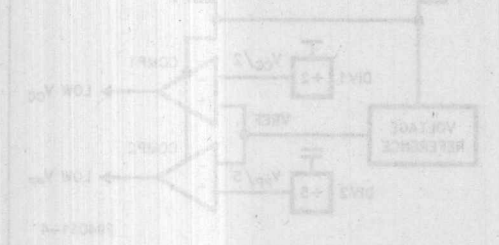


Figure 4. Low Power Detect Circuit

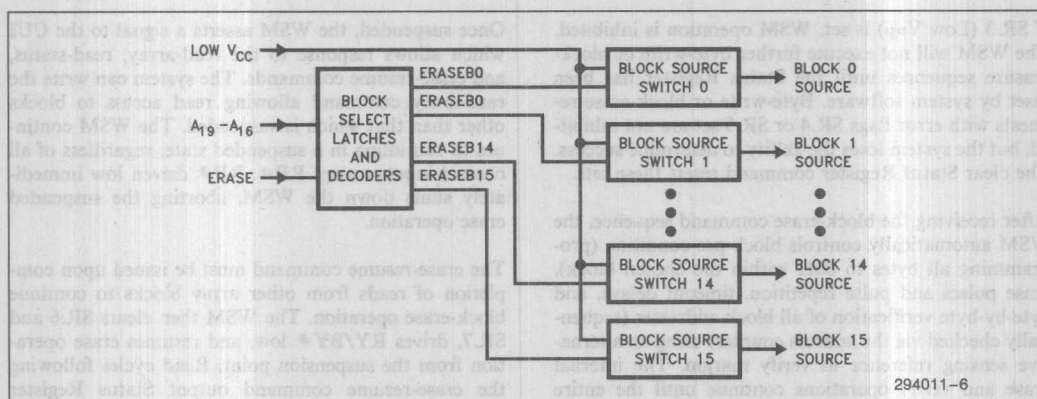


Figure 6. Array Erase Blocking

Block Erase

Block erasure is achieved by a two-step write sequence. The erase-setup code is written to the CUI in the first cycle. Erase confirm is written in the second cycle. The address supplied with the erase-confirm command is latched and decoded internally by the 28F008SA; erase is subsequently enabled in that block. The second WE# rising edge initiates the operation (WE#-controlled write).

The WSM triggers the high-voltage flash-erase switch connecting the 12V supply to the source of all bits in the specified block, while all wordlines are grounded. Figure 6 shows organization of the block source switches. Fowler-Nordheim tunneling results in simultaneous erasure of all bits in the selected block.

The block source switch controls the source voltage of all bits in a particular block. This circuit is shown in Figure 7. During block erase, M2 is off and M1 pulls the source to Vpp. When not in erase, M1 is off and M2

pulls the source to ground. The high-voltage latch formed by M4-M7 converts the low-voltage ERASE signal to a high-voltage signal that controls M1.

The tunneling that occurs during block erase requires only a small amount of current. However, the initial current required to charge the block's large source capacitance to the erase voltage is significant. M1 is sized to limit this current yet still apply sufficient source voltage to achieve fast block-erase time.

The LOWVCC signal protects the array from erasure when Vpp is at a high voltage but VCC is below the write/erase lockout voltage (VLKO). When this occurs, M3 pulls the block source to ground. The high-voltage latch is forced by M8 into the state that turns M1 off.

Vpp is continually monitored during all phases of the block-erase operation. If Vpp falls below the trip point of its high-voltage detect circuitry, erasure will not occur (or halts) and Status Register Vpp status (SR.3), block-erase status (SR.5) and WSM status (SR.7) bits are set to "1".

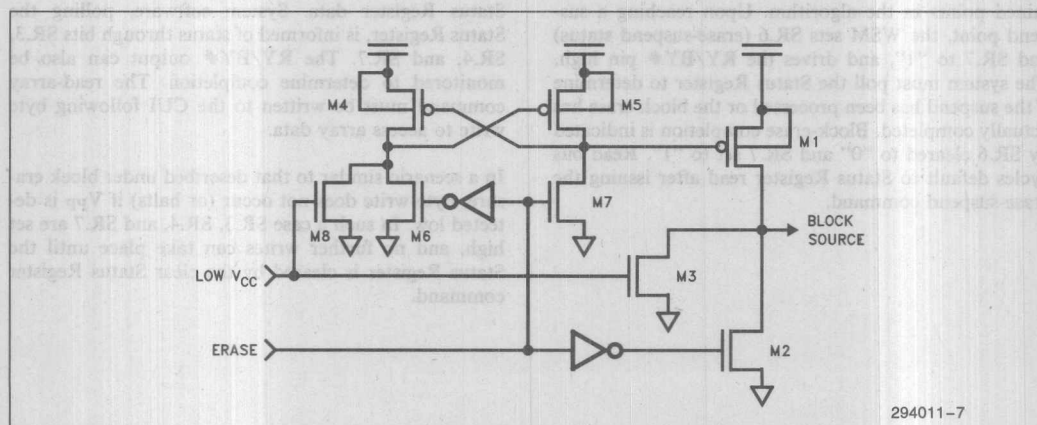


Figure 7. Block Source Switch

If SR.3 (Low V_{pp}) is set, WSM operation is inhibited. The WSM will not execute further byte-write or block-erase sequences until the Status Register has been reset by system software. Byte-write or block-erase requests with error flags SR.4 or SR.5 set are not inhibited, but the system loses the ability to determine success. The clear Status Register command resets these bits.

After receiving the block-erase command sequence, the WSM automatically controls block pre-condition (programming all bytes to 00H within the chosen block), erase pulses and pulse repetition, timeout delays, and byte-by-byte verification of all block addresses (sequentially checked via the address counter) using an alternative sensing reference to verify margin. The internal erase and verify operations continue until the entire block is erased. A read cycle applied to the part following the block-erase command sequence outputs Status Register contents; system software can poll the Status Register to determine when block erase is complete, and if it was successful. Alternately, the system can monitor RY/BY# until that output is driven high, and then poll the Status Register to determine success. Following block erasure, the device remains in Status Register read mode; a read-array command must be written to the device to access array data.

If the erase-setup command is followed with a command other than erase confirm, the device will not erase. The WSM sets both byte-write status and block-erase status bits in the Status Register to indicate an invalid sequence.

Erase Suspend/Resume

Erase suspend allows the system to interrupt block erase to read data from another array block. The ability to suspend erase and read data from another block offers the flexibility required for file system applications. Upon receiving the erase-suspend command, the CUI requests that the WSM pause at one of several predetermined points in the algorithm. Upon reaching a suspend point, the WSM sets SR.6 (erase-suspend status) and SR.7 to "1", and drives the RY/BY# pin high. The system must poll the Status Register to determine if the suspend has been processed or the block erase has actually completed. Block-erase completion is indicated by SR.6 cleared to "0" and SR.7 set to "1". Read bus cycles default to Status Register read after issuing the erase-suspend command.

Once suspended, the WSM asserts a signal to the CUI which allows response to the read-array, read-status, and erase-resume commands. The system can write the read-array command allowing read access to blocks other than that which is suspended. The WSM continues to run idling in a suspended state, regardless of all control inputs except RP#. RP# driven low immediately shuts down the WSM, aborting the suspended erase operation.

The erase-resume command must be issued upon completion of reads from other array blocks to continue block-erase operation. The WSM then clears SR.6 and SR.7, drives RY/BY# low, and resumes erase operation from the suspension point. Read cycles following the erase-resume command output Status Register data.

Byte Write

Byte write follows a flow similar to block erase. The byte-write setup command is first written to the CUI. A second write cycle loads address and data latches. The rising edge of the second WE# pulse requests that the WSM initiate activity, applying high voltage to the gates and drains of all bits to be written. Unlike block erase, byte write will proceed regardless of what data is applied on the second CUI write cycle; however, applying data FFH does not modify memory contents.

Like block erase, the WSM controls program pulses and pulse repetition, timeout delays and byte verification. Byte write and verify (with alternate sensing reference and internally-generated verify voltage) continue until the byte is written. Internal byte-write verify checks that all bits written to zero have been correctly modified; it does not check bits specified as one. Byte write cannot change existing zeros to ones; this can only be accomplished by erase.

Read bus cycles following byte write operations output Status Register data. System software, polling the Status Register, is informed of status through bits SR.3, SR.4, and SR.7. The RY/BY# output can also be monitored to determine completion. The read-array command must be written to the CUI following byte write to access array data.

In a scenario similar to that described under block erasure, byte write does not occur (or halts) if V_{pp} is detected low. In such a case SR.3, SR.4, and SR.7 are set high, and no further writes can take place until the Status Register is cleared by the clear Status Register command.

DEVICE CHARACTERIZATION

AC and DC Parameters

Figures 9 through 24 show graphs of several device parameters as a function of temperature and supply voltage. The graphs illustrate that the 28F008SA has significant margin to data sheet specifications.

In particular, note Figure 9 which shows typical read performance t_{AVQV} (t_{ACC}) of the 28F008SA as a function of V_{CC} and ambient temperature. t_{ELQV} (t_{CE}) in Figure 10 and t_{GLQV} (t_{OE}) in Figure 11 are also of particular interest. Access times t_{AVQV} , t_{ELQV} , and t_{GLQV} are specified and tested with an output load of 100 pF; additional output load capacitance slows device operation.

Table 1 shows typical supply currents at room temperature for several operating modes.

Table 1. RMS Current Values

Mode	I_{CC} ($V_{CC} = 5.0V$, CMOS Inputs)	I_{PP} ($V_{PP} = 12V$)
Read	20 mA	100 μA
Byte Write	10 mA	12 mA
Block Erase	10 mA	12 mA
Standby	40 μA	100 μA
Deep Power-Down	0.20 μA	0.07 μA

Energy/Power Consumption

The system designer is primarily concerned with power consumption during block erase and byte write. Typical curves for I_{CC} and I_{PP} during block erase are shown in Figure 25. I_{CC} and I_{PP} for byte write are illustrated in Figure 26.

Byte-Write/Block-Erase Times

The 28F008SA advances byte-write and block-erase performance compared to previous flash memories. The on-chip algorithm is improved over the 28F001BX to take advantage of process enhancements. This improvement is most apparent when compared to first-generation flash parts with externally controlled algorithms. First-generation device times shown in Table 2 assume optimal system overhead, and as such are absolute best case.

Table 2. Byte-Write and Block-Erase Performance vs Previous Devices

Device	Byte-Write Time	Block-Erase Time/ # Bytes	Erase Time per Kbyte
Second-Generation Flash Memory Devices⁽¹⁾			
28F008SA	9 μs	1.5s/64K	23 ms
28F001BX	18 μs	3.8s/112K	34 ms
		2.1s/8K	256 ms
		2.1s/4K	513 ms
First-Generation Flash Memory Devices⁽²⁾			
28F020	16.5 μs	6.8s/256K	27 ms
28F010	16.5 μs	3.9s/128K	30 ms
28F512	16.5 μs	2.4s/64K	37 ms
28F256A	16.5 μs	1.6s/32K	51 ms

NOTES:

1. Typical measured time.
2. Times calculated based on typical erase and precondition pulse requirements, with minimum write timings. Calculations are described in Figure 8.

Figure 27 shows block-write and block-erase times at 0°C and 70°C over cycling.

DEVICE RELIABILITY

Byte-Write/Block-Erase Cycling

One of the most important reliability aspects of the 28F008SA is its capability of 100,000 write/erase cycles per block. Destructive oxide breakdown has been a limiting factor in extended cycling of thin-oxide EEPROMs. Intel's ETOX III flash memory technology extends cycling performance through:

- Improved tunnel-oxide processing that increases charge-carrying capability tenfold.
- Significantly reduced oxide area under stress that minimizes probability of oxide defects in the region.
- Reduced oxide stress due to a lower peak electric field (lower erase voltage than EEPROM).

Reliable byte-write/block-erase cycling requires proper selection of the maximum erase threshold voltage (V_t), and maintenance of a tight distribution. Maximum erase V_t is set to 3.4V via the internal block-erase algorithm and verify circuits. Tight erase V_t distribution gives an order of magnitude of erase-time margin to the fastest erasing cell, with virtually identical erase V_t distributions at 1 and 10,000 cycles (Figure 28). Program V_t distribution is similarly consistent over cycling (Figure 29).

ER-27 During byte write and erase pre-conditioning. First, only one of the two row decoders is active at any time (selected by A_{19}). Rows in the other two quadrants are grounded. Secondly, two separate internally-switched voltages supply the left and right quadrants. Only one supply (selected by block address A_{18}) is switched to programming voltages while the other remains at read voltages. This A_{19} - A_{18} row decoding ensures that during byte write, 12 of the 16 blocks have a gate voltage below that required for programming.

Data Protection

The 28F008SA offers protection against accidental block erasure or byte write during power transitions. Internal circuitry creates a device insensitive to V_{pp}/V_{CC} supply power-up sequencing. $V_{pp} \leq V_{pPL}$ locks out byte-write and block-erase circuits. $V_{CC} \leq V_{LKO}$ disables CUI command writes, resets the CUI to array read mode, and holds the WSM inactive. The system designer must still guard against spurious command writes for $V_{CC} > V_{LKO}$ when $V_{pp} > V_{pPL}$.

Several strategies are available to prevent data modification in the 28F008SA. The CUI provides a degree of software write protection since memory alteration occurs only after successful completion of a two-step write sequence. $WE\#$ and $CE\#$ must both go active to perform this sequence; driving either high inhibits command/data writes. Secondly, the system can place the device in deep power-down mode ($RP\# = V_{IL}$) to disable command writes, reset the CUI to array read

mode, or switch it to V_{pPL} only when memory updates are required. Since byte-write and block-erase circuits are disabled by $V_{pp} \leq V_{pPL}$, V_{pp} switching adds another level of data security.

SUMMARY

The 28F008SA is the first flash memory with features optimized for solid-state systems and file storage. These features include symmetrical block-erase, automation of byte write and block erase, erase suspend for data read, a reset/power-down mode, a write/erase Status Register, and a dedicated RY/BY# status pin. With simple microprocessor interfacing and software command sequences, the 28F008SA is the non-volatile storage solution of choice for today's designs.

OTHER REFERENCES

Related documents of interest to readers of this engineering report:

- 28F008SA Data Sheet (order #290429)
- 28F008SA-L Data Sheet (order #290435)
- AP-359 "28F008SA Hardware Interfacing" (order #292094)
- AP-360 "28F008SA Software Drivers" (order #292095)
- AP-364 "28F008SA Automation and Algorithms" (order #292099)
- ER-28 "ETOX III Flash Memory Technology" (order #294012)

SUPPLEMENTARY INFORMATION

FORMULA:

b = # bytes in a block (256K, 128K, 64K, 32K)
 n = # of erase pulses required (90 pulses)
 w = time for a write cycle (150 ns, t_{AVAV})
 v = time to verify (6055 ns, $t_{WHGL} + t_{GLQV}$)
 p = program pulse width (10 μ s, t_{WHWH1}) one pulse programming assumed
 e = erase pulse width (10 ms, t_{WHWH2})

Precondition and precondition verify time is:
 $b(2w + p + v)$

Erase/verify, each loop where some byte does not pass verify:

$$(n - 1)(2w + e + v)$$

Last erase pulse:

$$(1)(2w + e)$$

Passing erase-verify, all bytes:

$$b(w + v)$$

Total time can be summarized as:

$$b(3w + p + 2v) + n(2w + e + v) - (v)$$

or substituting in times for write, verify, program and erase pulse widths:

$$b(22.56 \times 10^{-6}) + n(10.006355 \times 10^{-3}) - (6.055 \times 10^{-6}) \text{ Seconds}$$

Figure 8. Erase Time Calculations for First-Generation Flash Memories

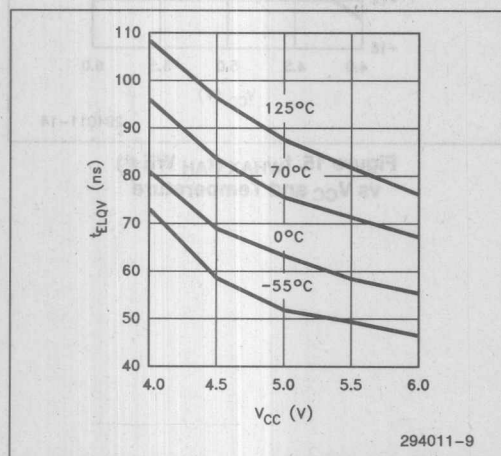


Figure 10. t_{ELQV} (t_{CE}) vs V_{CC} and Temperature

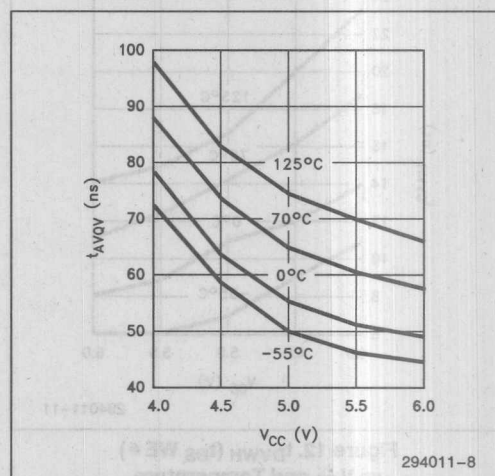


Figure 9. t_{AVQV} (t_{ACC}) vs V_{CC} and Temperature

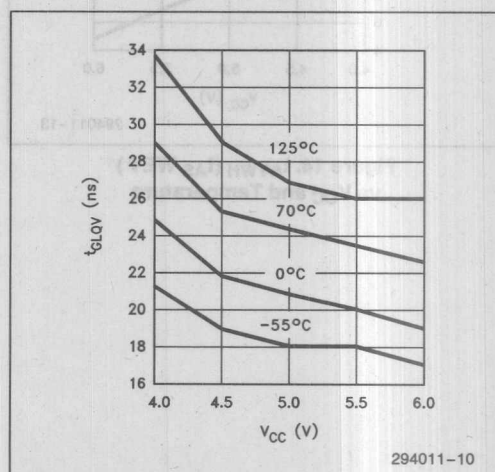


Figure 11. t_{GLQV} (t_{OE}) vs V_{CC} and Temperature

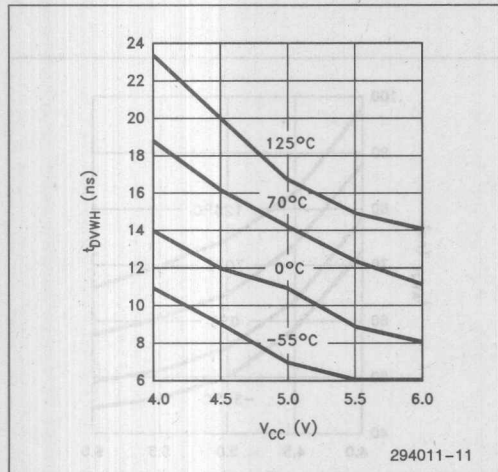


Figure 12. t_{DVWH} (t_{DS} WE#)
vs V_{CC} and Temperature

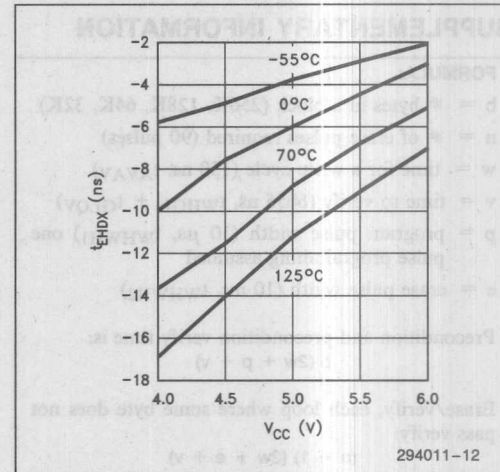


Figure 13. t_{EHDX} (t_{DH} CE#)
vs V_{CC} and Temperature

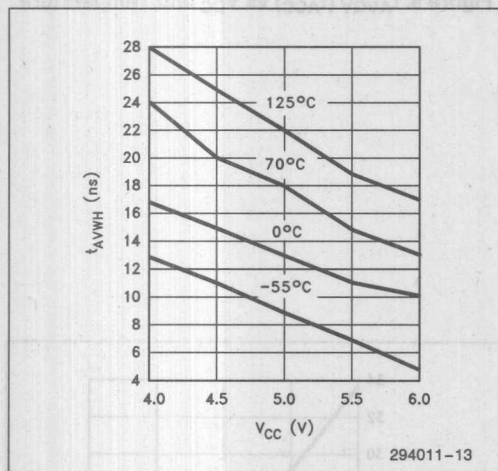


Figure 14. t_{AVWH} (t_{AS} WE#)
vs V_{CC} and Temperature

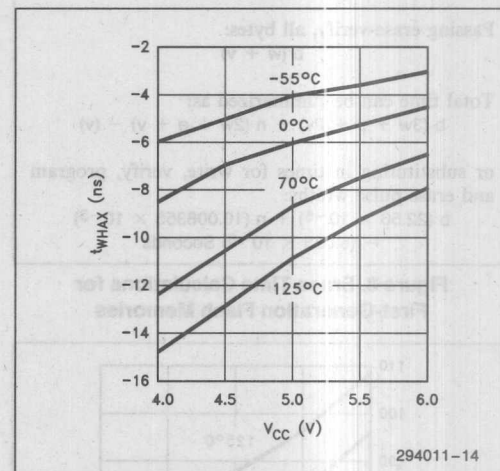


Figure 15. t_{WHAX} (t_{AH} WE#)
vs V_{CC} and Temperature

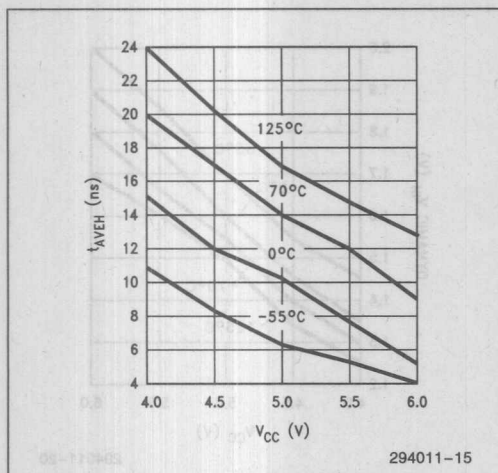


Figure 16. t_{AVEH} (t_{AS} CE#) vs V_{CC} and Temperature

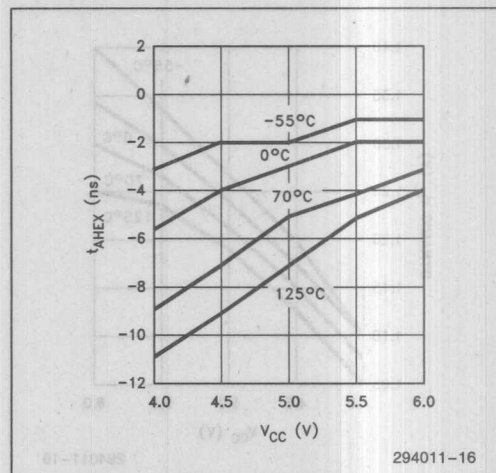


Figure 17. t_{AHEX} (t_{AH} CE#) vs V_{CC} and Temperature

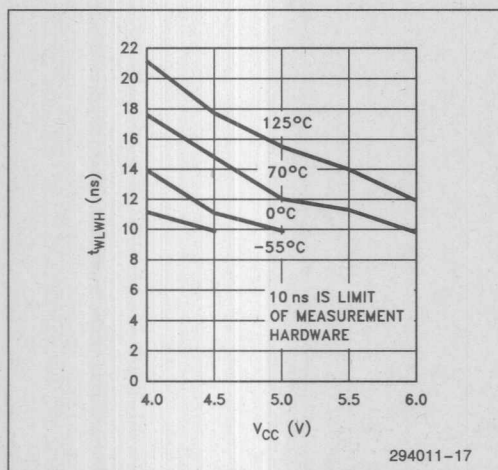


Figure 18. t_{WLWH} (t_{WP}) vs V_{CC} and Temperature

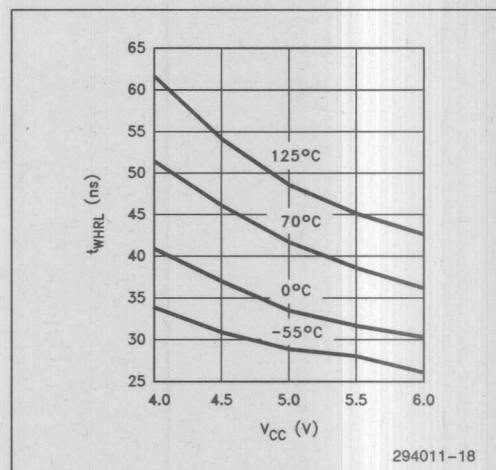


Figure 19. t_{WHRL} vs V_{CC} and Temperature

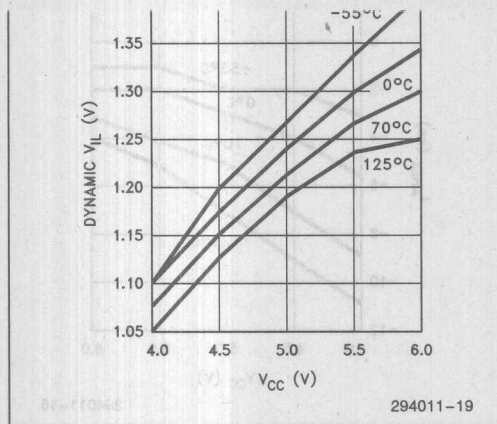


Figure 20. Dynamic V_{IL} vs V_{CC} and Temperature

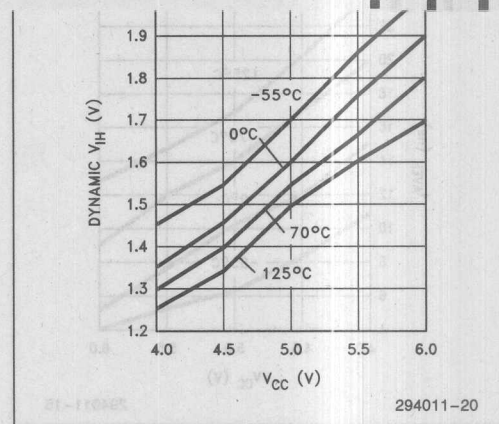


Figure 21. Dynamic V_{IH} vs V_{CC} and Temperature

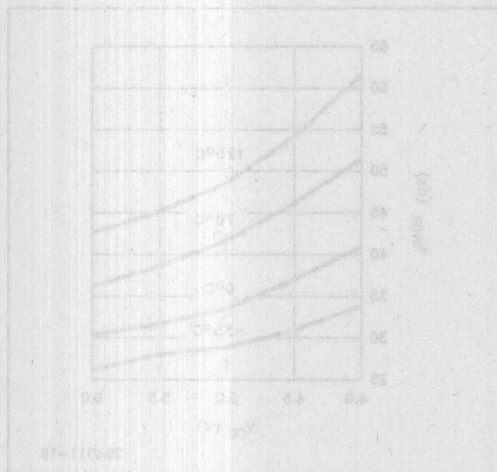


Figure 18. Power vs V_{CC} and Temperature

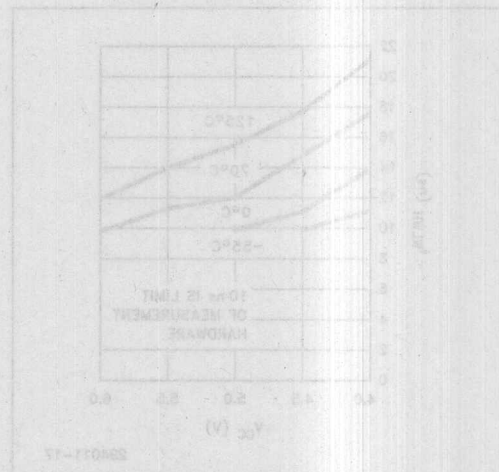


Figure 16. I_{OH} vs V_{CC} and Temperature

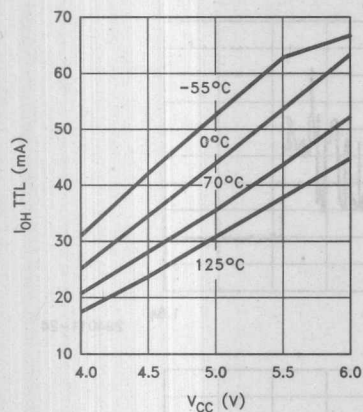


Figure 22. I_{OH} TTL vs V_{CC} and Temperature
($V_{OH} = 2.4V$)

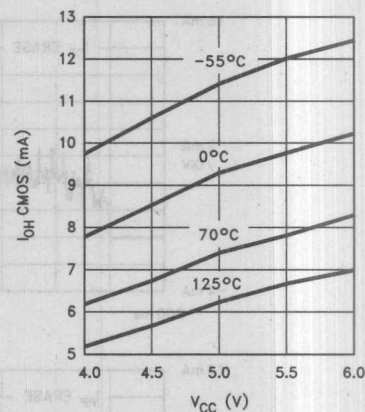


Figure 23. I_{OH} CMOS vs V_{CC} and Temperature
($V_{OH} = V_{CC} - 0.4V$)

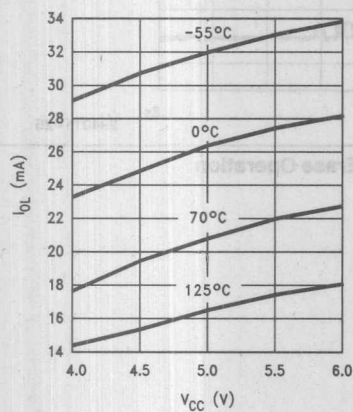
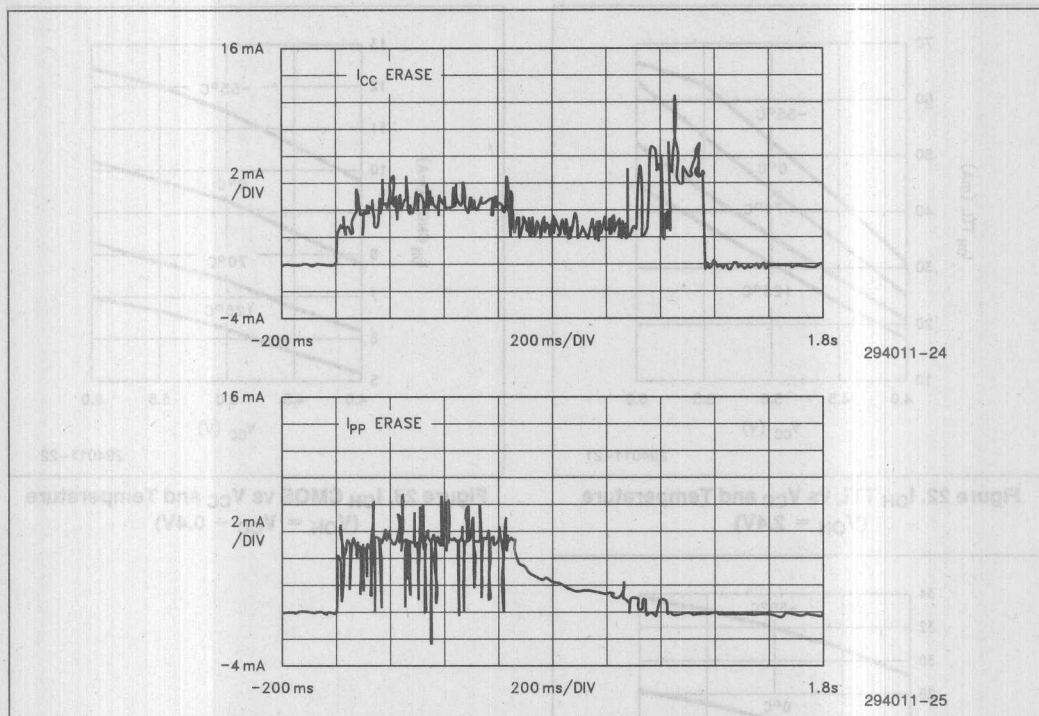


Figure 24. I_{OL} vs V_{CC} and Temperature
($V_{OL} = 0.45V$)

Figure 25. I_{CC} and I_{PP} under Block-Erase Operation

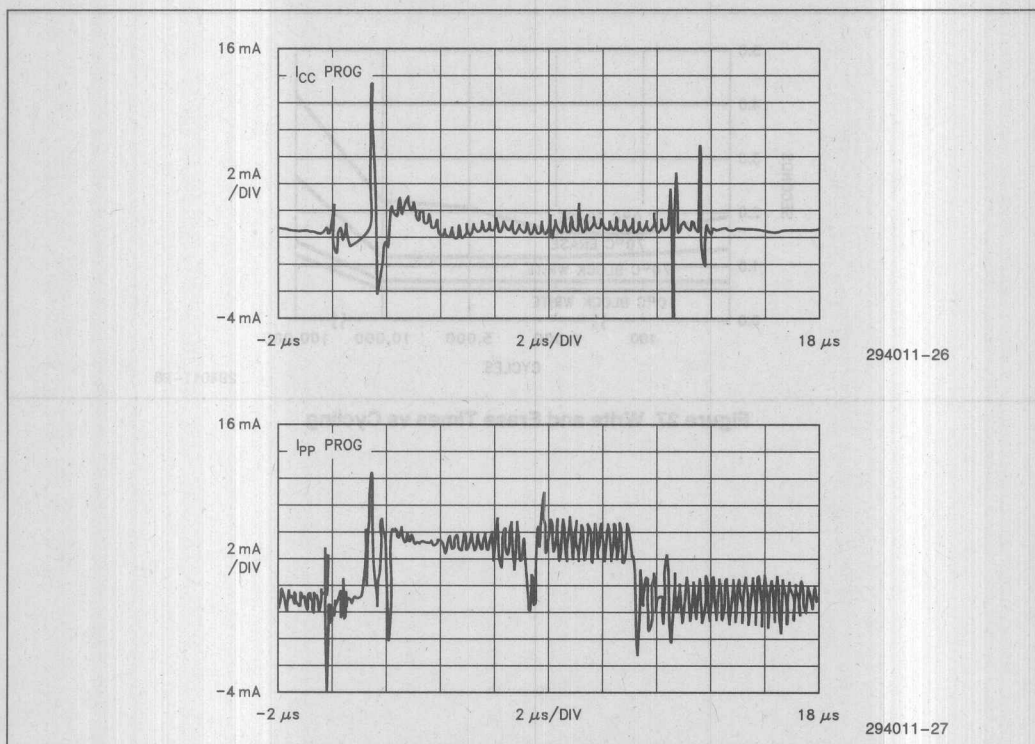


Figure 26. I_{CC} and I_{PP} under Byte-Write Operation

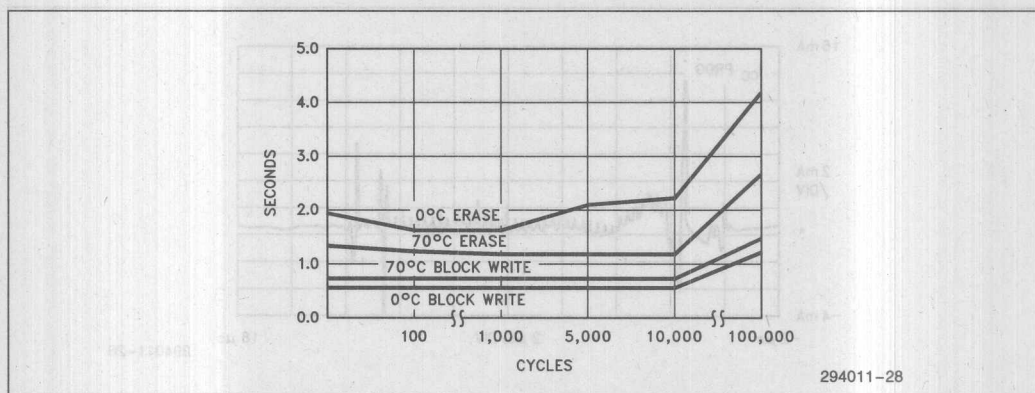


Figure 27. Write and Erase Times vs Cycling

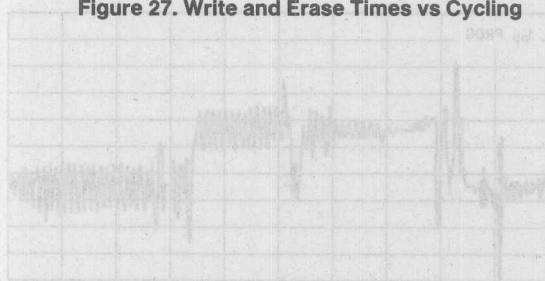


Figure 28. I/O and I/O under Write Operation

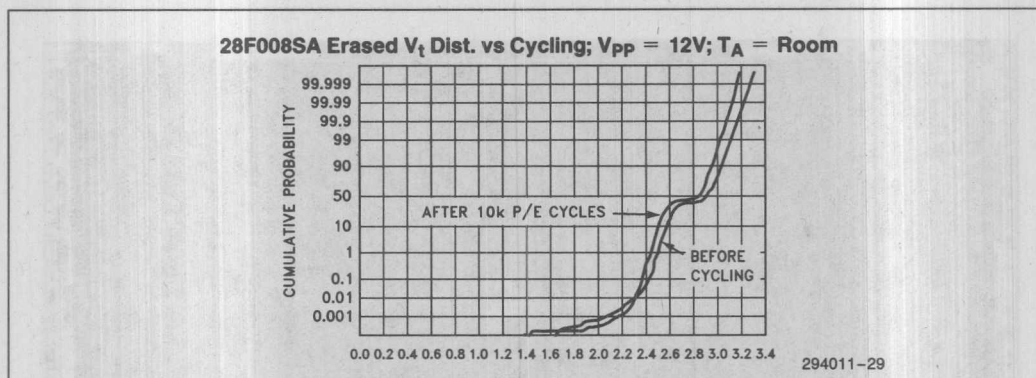


Figure 28. Erase V_t vs Cycles

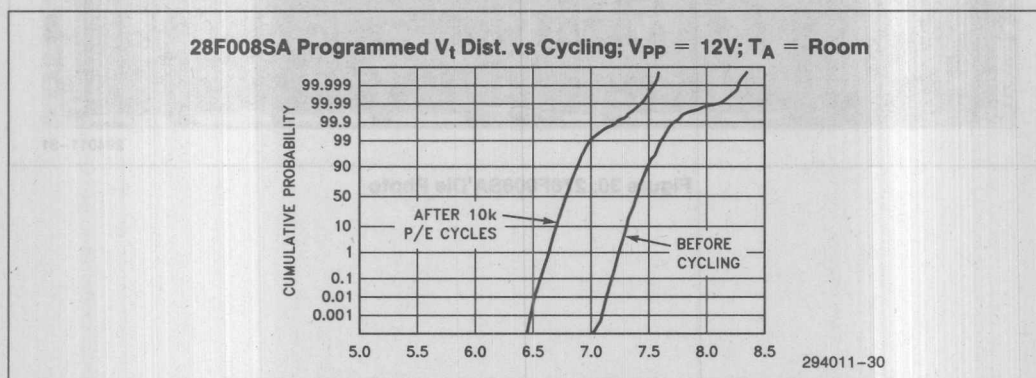


Figure 29. Program V_t vs Cycles

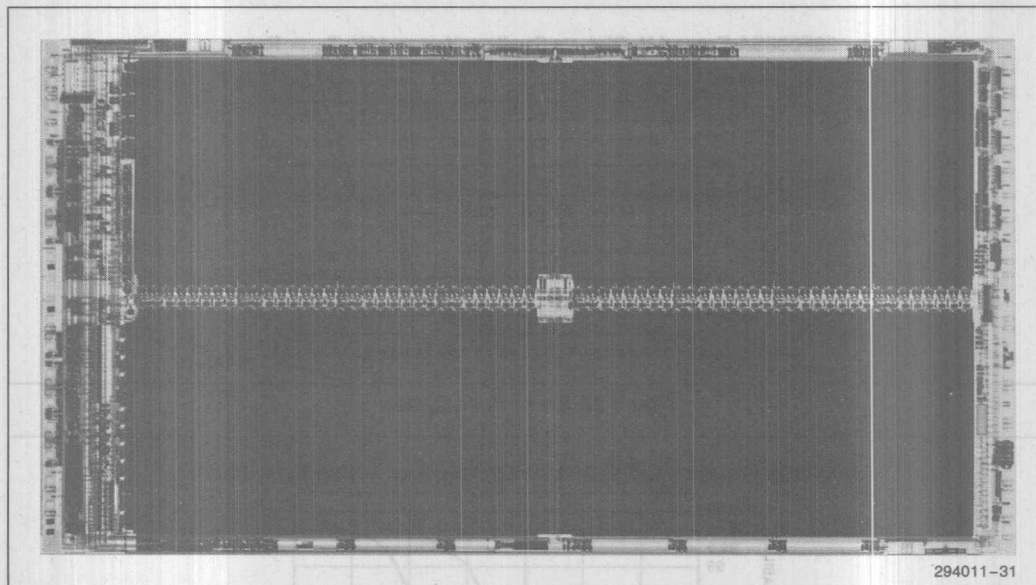
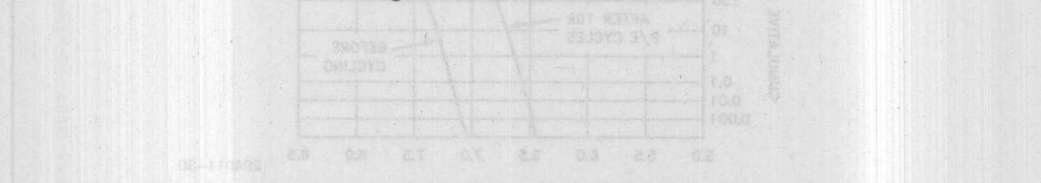


Figure 30. 278F008SA Die Photo



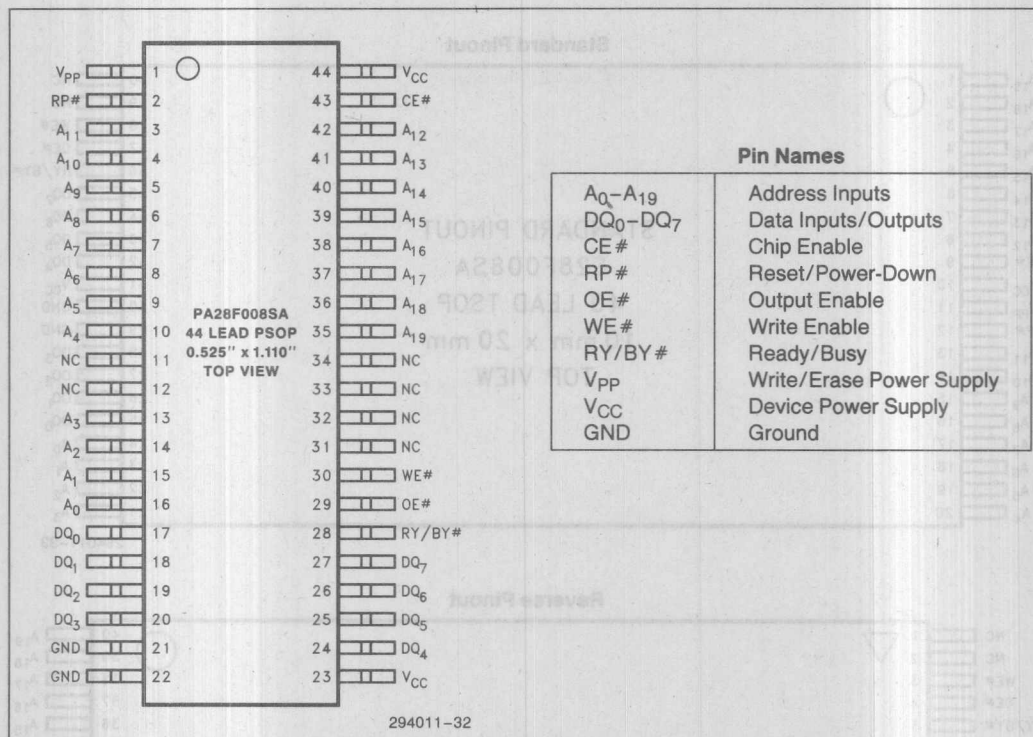


Figure 31. PSOP Lead Configuration

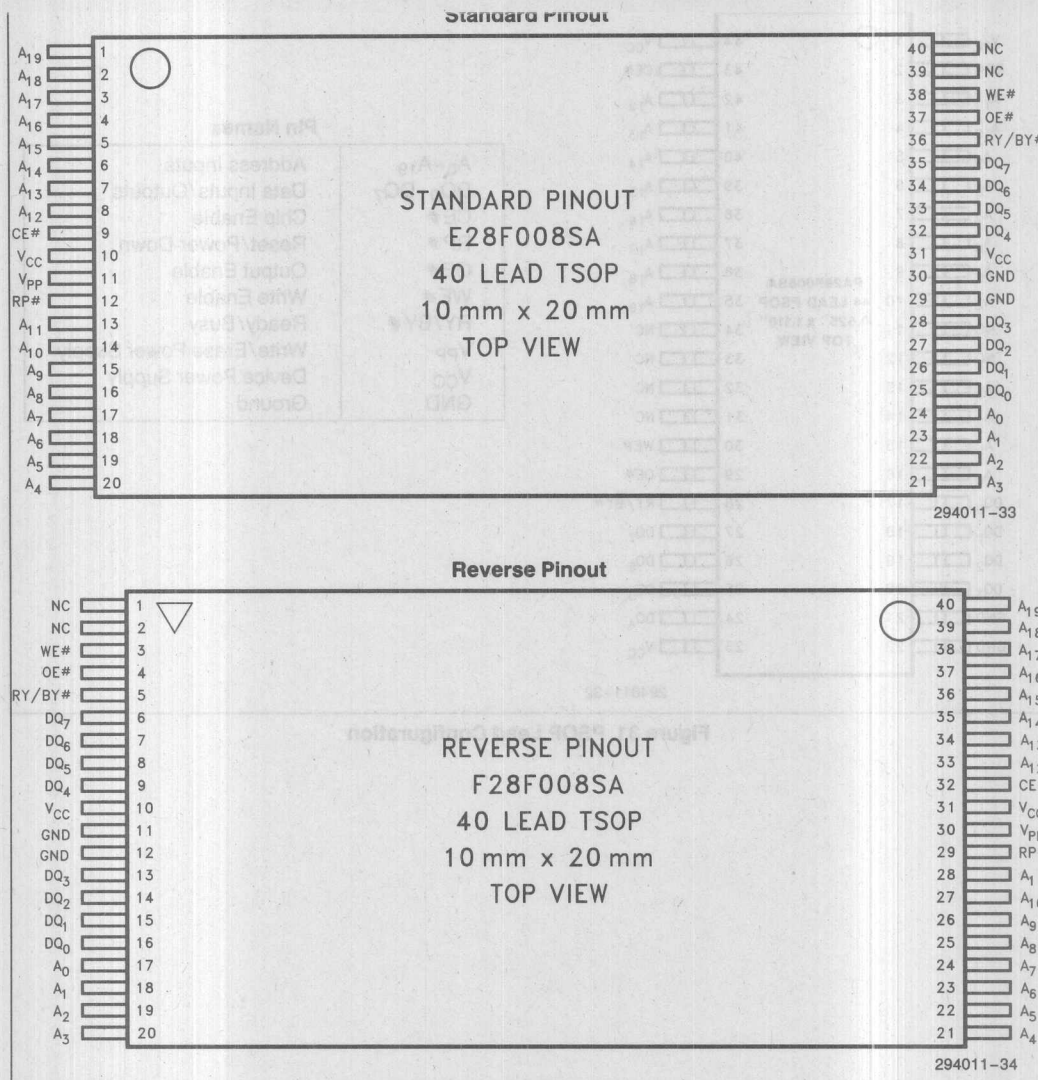


Figure 32. TSOP Lead Configuration

Wordlines are numbered sequentially from top to bottom. Addresses A₉–A₀ sequentially decode wordlines: block address A₁₉ selects between upper and lower row-decoder. Wordlines 0–1023 serve the left and right quadrants at top of device; 1024–2047 serve the lower quadrants.

Table 3. Row Address Bitmap

A ₁₉	A ₉	A ₈	A ₇	A ₆	A ₅	A ₄	A ₃	A ₂	A ₁	A ₀	Wordline
0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	1	1
0	0	0	0	0	0	0	0	0	1	0	2
0	0	0	0	0	0	0	0	0	1	1	3
0	0	0	0	0	0	0	0	1	0	0	4
0	0	0	0	0	0	0	0	1	0	1	5
0	0	0	0	0	0	0	0	1	1	0	6
0	0	0	0	0	0	0	0	1	1	1	7
0	0	0	0	0	0	0	1	0	0	0	8
0	0	0	0	0	0	0	1	0	0	1	9
0	0	0	0	0	0	0	1	0	1	0	10
0	0	0	0	0	0	0	1	0	1	1	11
0	0	0	0	0	0	0	1	1	0	0	12
0	0	0	0	0	0	0	1	1	0	1	13
0	0	0	0	0	0	0	1	1	1	0	14
0	0	0	0	0	0	0	1	1	1	1	15
0	0	0	0	0	0	1	0	0	0	0	16
0	0	0	0	0	0	1	•	•	•	•	•
0	0	0	0	0	0	1	1	1	1	1	31
0	•	•	•	•	•	•	•	•	•	•	•
0	1	1	1	1	1	1	0	0	0	0	1008
0	1	1	1	1	1	1	•	•	•	•	•
0	1	1	1	1	1	1	1	1	1	1	1023
1	0	0	0	0	0	0	0	0	0	0	1024
1	0	0	0	0	0	0	•	•	•	•	•
1	0	0	0	0	0	0	1	1	1	1	1039
1	0	0	0	0	0	1	0	0	0	0	1040
1	•	•	•	•	•	•	•	•	•	•	•
1	1	1	1	1	1	1	1	1	1	1	2047

Blocks are numbered sequentially right to left, top to bottom. Columns within a block are numbered left to right. I/Os within a block are numbered from 0–7 left to right.

Table 4. Block Address Bitmap

A ₁₉	A ₁₈	A ₁₇	A ₁₆	Block	Columns
0	0	0	0	0	0–511
0	0	0	1	1	512–1023
0	0	1	0	2	1024–1535
0	0	1	1	3	1536–2047
0	1	0	0	4	2048–2559
0	1	0	1	5	2560–3071
0	1	1	0	6	3072–3583
0	1	1	1	7	3584–4095
1	0	0	0	8	4096–4607
1	0	0	1	9	4608–5119
1	0	1	0	10	5120–5631
1	0	1	1	11	5632–6143
1	1	0	0	12	6144–6655
1	1	0	1	13	6656–7167
1	1	1	0	14	7168–7679
1	1	1	1	15	7680–8191

Columns are numbered from left to right across the top quadrants, and left to right across the bottom quadrants. Addresses A₁₅–A₁₀ sequentially count columns. Columns are listed for block 0; other blocks are counted similarly.

Table 5. Column Address Bitmap

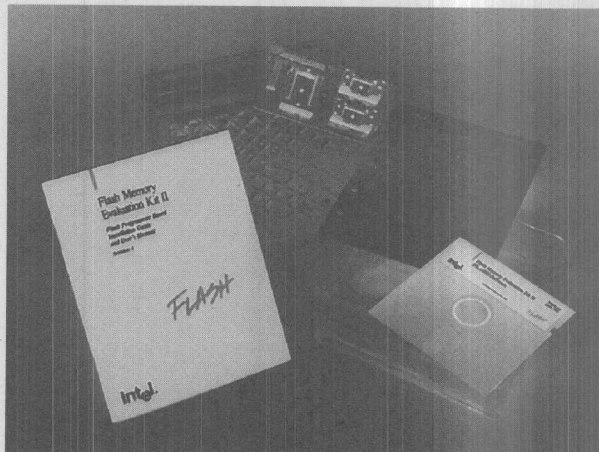
A ₁₅	A ₁₄	A ₁₃	A ₁₂	A ₁₁	A ₁₀	Column In							
						I/O ₀	I/O ₁	I/O ₂	I/O ₃	I/O ₄	I/O ₅	I/O ₆	I/O ₇
0	0	0	0	0	0	0	64	128	192	256	320	384	448
0	0	0	0	0	1	1	65	129	193	257	321	385	449
0	0	0	0	1	0	2	66	130	194	258	322	386	450
0	0	0	0	1	1	3	67	131	195	259	323	387	451
0	0	0	1	0	0	4	68	132	196	260	324	388	452
•	•	•	•	•	•	•	•	•	•	•	•	•	•
1	1	1	1	1	0	62	126	190	254	318	382	446	510
1	1	1	1	1	1	63	127	191	255	319	383	447	511

REVISION HISTORY

-002 — Renamed PWD as RP# to match JEDEC conventions.

Intel 28F008SA FlashFile™ Memory Evaluation Module D, FLASHEVAL4 Product Brief

- Kit Contents**
- 28F008SA Adapter Board with:
 - 40-ld TSOP socket ('E', standard pinout)
 - 40-ld TSOP socket ('F', reverse pinout)
 - 44-ld PSOP socket
 - 5.25" floppy disk with iFlash2 Software (Version 2.4)
 - Technical documentation describing Intel's 28F008SA FlashFile™ Memory
 - Flash Memory Evaluation Kit II Installation Guide and User's Manual with 28F008SA Adapter Board installation instructions
 - Manual vacuum wand
 - Registration card



Intel's 28F008SA FlashFile™ Memory Evaluation Module provides system designers with a cost-effective prototyping tool for writing and erasing the 28F008SA FlashFile Memory. This evaluation module is a hardware adapter board upgrade to the Intel Flash Memory Evaluation Kit II (D, FLASHEVAL2) which supports the 28F008SA in 40-lead TSOP (standard and reverse pinouts) and 44-lead PSOP packages.

Kit Description

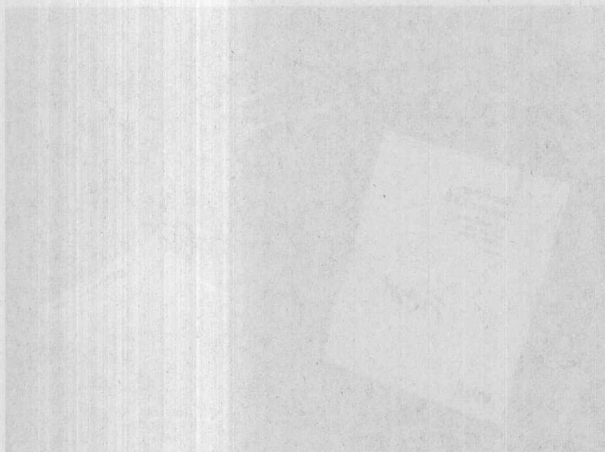
The 28F008SA Evaluation Module, used with Intel's Flash Memory Evaluation Kit, provides the hardware, software and system interface necessary to evaluate and integrate Intel's 8Mbit Flash Memory into your next design.

The module provides instructions to install the 28F008SA adapter board. Technical documentation includes 28F008SA datasheets, engineering reports and application notes. Together, they provide a complete description of the technology and important design considerations.

FlashFile is a trademark of Intel Corporation

intel.

Intel 28F0082A FlashFile™ Memory Evaluation Module D, FLASHVAL-4 Product Brief



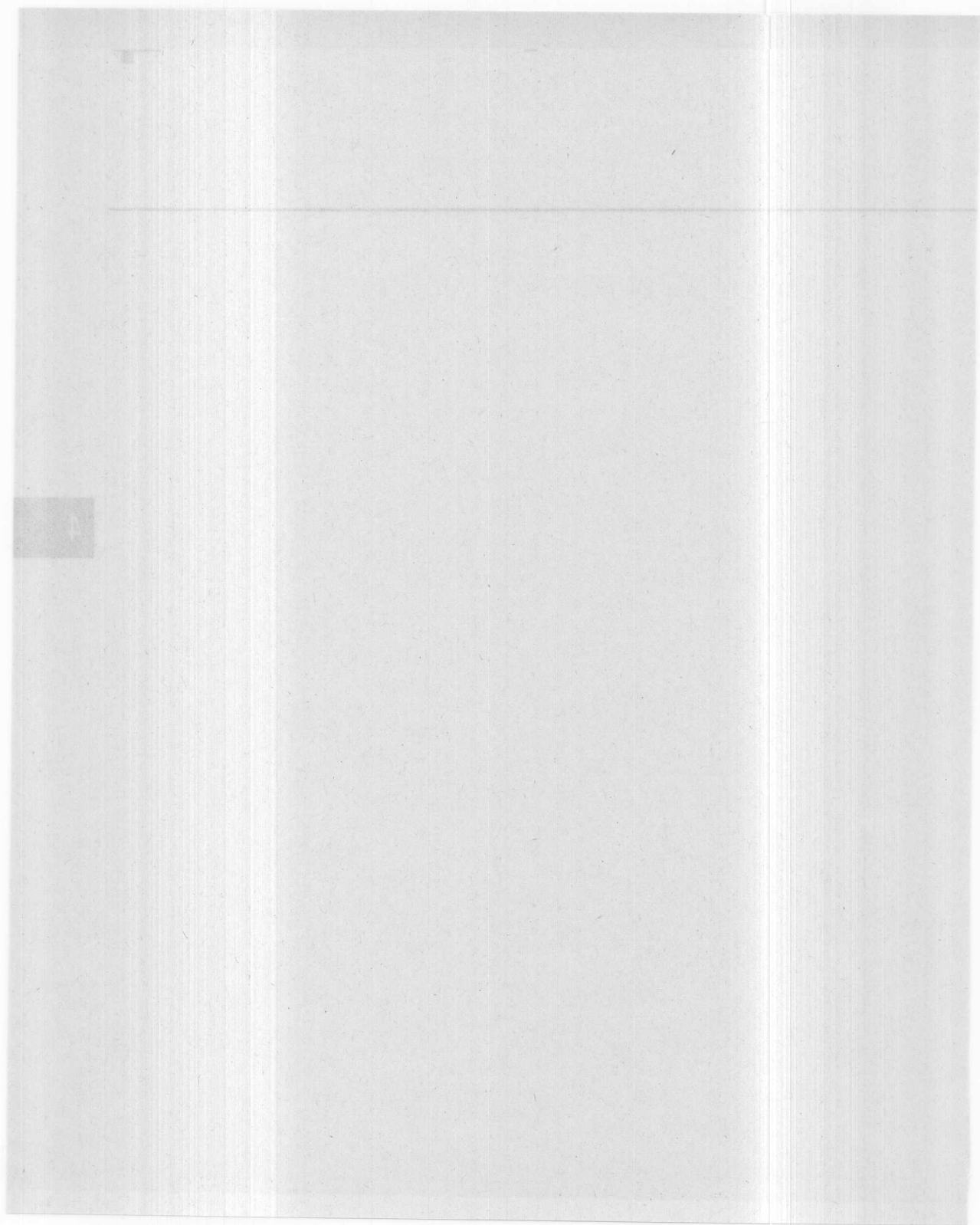
The Intel 28F0082A FlashFile™ Memory Evaluation Module D is a single-board evaluation module for the Intel 28F0082A FlashFile™ Memory Evaluation Module D. This evaluation module is a single-board evaluation module for the Intel 28F0082A FlashFile™ Memory Evaluation Module D. It is designed to evaluate the performance of the Intel 28F0082A FlashFile™ Memory Evaluation Module D in a single-board environment. The module is designed to be used with a single-board computer system. It is designed to be used with a single-board computer system. It is designed to be used with a single-board computer system.

Kit Description

The 28F0082A FlashFile™ Memory Evaluation Module D is a single-board evaluation module for the Intel 28F0082A FlashFile™ Memory Evaluation Module D. It is designed to evaluate the performance of the Intel 28F0082A FlashFile™ Memory Evaluation Module D in a single-board environment. The module is designed to be used with a single-board computer system. It is designed to be used with a single-board computer system. It is designed to be used with a single-board computer system.

- Kit B: 28F0082A Adapter Board with:
 - 40-pin TSOP socket
 - 16-pin TSOP socket
 - 40-pin TSOP socket
 - 16-pin TSOP socket
 - 40-pin TSOP socket
- Kit C: 28F0082A Adapter Board with 16-pin TSOP socket
- Kit D: 28F0082A Adapter Board with 40-pin TSOP socket
- Kit E: 28F0082A Adapter Board with 16-pin TSOP socket
- Kit F: 28F0082A Adapter Board with 40-pin TSOP socket
- Kit G: 28F0082A Adapter Board with 16-pin TSOP socket
- Kit H: 28F0082A Adapter Board with 40-pin TSOP socket
- Kit I: 28F0082A Adapter Board with 16-pin TSOP socket
- Kit J: 28F0082A Adapter Board with 40-pin TSOP socket
- Kit K: 28F0082A Adapter Board with 16-pin TSOP socket
- Kit L: 28F0082A Adapter Board with 40-pin TSOP socket
- Kit M: 28F0082A Adapter Board with 16-pin TSOP socket
- Kit N: 28F0082A Adapter Board with 40-pin TSOP socket
- Kit O: 28F0082A Adapter Board with 16-pin TSOP socket
- Kit P: 28F0082A Adapter Board with 40-pin TSOP socket
- Kit Q: 28F0082A Adapter Board with 16-pin TSOP socket
- Kit R: 28F0082A Adapter Board with 40-pin TSOP socket
- Kit S: 28F0082A Adapter Board with 16-pin TSOP socket
- Kit T: 28F0082A Adapter Board with 40-pin TSOP socket
- Kit U: 28F0082A Adapter Board with 16-pin TSOP socket
- Kit V: 28F0082A Adapter Board with 40-pin TSOP socket
- Kit W: 28F0082A Adapter Board with 16-pin TSOP socket
- Kit X: 28F0082A Adapter Board with 40-pin TSOP socket
- Kit Y: 28F0082A Adapter Board with 16-pin TSOP socket
- Kit Z: 28F0082A Adapter Board with 40-pin TSOP socket

intel



28F400BX-T/B, 28F004BX-T/B

4 MBIT (256K x16, 512K x8) BOOT BLOCK FLASH MEMORY FAMILY

- **x8/x16 Input/Output Architecture**
 - 28F400BX-T, 28F400BX-B
 - For High Performance and High Integration 16-bit and 32-bit CPUs
- **x8-only Input/Output Architecture**
 - 28F004BX-T, 28F004BX-B
 - For Space Constrained 8-bit Applications
- **Optimized High Density Blocked Architecture**
 - One 16-KB Protected Boot Block
 - Two 8-KB Parameter Blocks
 - One 96-KB Main Block
 - Three 128-KB Main Blocks
 - Top or Bottom Boot Locations
- **Extended Cycling Capability**
 - 100,000 Block Erase Cycles
- **Automated Word/Byte Write and Block Erase**
 - Command User Interface
 - Status Registers
 - Erase Suspend Capability
- **SRAM-Compatible Write Interface**
- **Automatic Power Savings Feature**
 - 1 mA Typical I_{CC} Active Current in Static Operation
- **Very High-Performance Read**
 - 60/80 ns Maximum Access Time
 - 30/40 ns Maximum Output Enable Time
- **Low Power Consumption**
 - 20 mA Typical x8 Active Read Current
 - 25 mA Typical x16 Active Read Current
- **Reset/Deep Power-Down Input**
 - 0.2 μ A I_{CC} Typical
 - Acts as Reset for Boot Operations
- **Extended Temperature Operation**
 - -40°C to +85°C
- **Write Protection for Boot Block**
- **Hardware Data Protection Feature**
 - Erase/Write Lockout During Power Transitions
- **Industry Standard Surface Mount Packaging**
 - 28F400BX: JEDEC ROM Compatible 44-Lead PSOP
 - 56-Lead TSOP
 - 28F004BX: 40-Lead TSOP
- **12V Word/Byte Write and Block Erase**
 - V_{pp} = 12V \pm 5% Standard
 - V_{pp} = 12V \pm 10% Option
- **ETOX III Flash Technology**
 - 5V Read

Intel's 4-Mbit Flash Memory Family is an extension of the Boot Block Architecture which includes block-selective erasure, automated write and erase operations and standard microprocessor interface. The 4-Mbit Flash Memory Family enhances the Boot Block Architecture by adding more density and blocks, x8/x16 input/output control, very high speed, low power, an industry standard ROM compatible pinout and surface mount packaging. The 4-Mbit flash family is an easy upgrade from Intel's 2-Mbit Boot Block Flash Memory Family.

The Intel 28F400BX-T/B are 16-bit wide flash memory offerings. These high density flash memories provide user selectable bus operation for either 8-bit or 16-bit applications. The 28F400BX-T and 28F400BX-B are 4,194,304-bit non-volatile memories organized as either 524,288 bytes or 262,144 words of information. They are offered in 44-Lead plastic SOP and 56-Lead TSOP packages. The x8/x16 pinout conforms to the industry standard ROM/EPROM pinout.

The Intel 28F004BX-T/B are 8-bit wide flash memories with 4,194,304 bits organized as 524,288 bytes of information. They are offered in a 40-Lead TSOP package, which is ideal for space-constrained portable systems.

These devices use an integrated Command User Interface (CUI) and Write State Machine (WSM) for simplified word/byte write and block erasure. The 28F400BX-T/28F004BX-T provide block locations compatible with Intel's MCS-186 family, 80286, i386™, i486™, i860™ and 80960CA microprocessors. The 28F400BX-B/28F004BX-B provide compatibility with Intel's 80960KX and 80960SX families as well as other embedded microprocessors.

The boot block includes a data protection feature to protect the boot code in critical applications. With a maximum access time of 60 ns, these 4-Mbit flash devices are very high performance memories which interface at zero-wait-state to a wide range of microprocessors and microcontrollers. A deep power-down mode lowers the total V_{CC} power consumption to 1 μ W. This is critical in handheld battery powered systems. For very low power applications using a 3.3V supply, refer to the Intel 28F400BX-TL/BL, 28F004BX-TL/BL 4-Mbit Boot Block Flash Memory Family datasheet.

Manufactured on Intel's 0.8 micron ETOX III process, the 4-Mbit flash memory family provides world class quality, reliability and cost-effectiveness at the 4-Mbit density level.

1.0 PRODUCT FAMILY OVERVIEW

Throughout this datasheet the 28F400BX refers to both the 28F400BX-T and 28F400BX-B devices and 28F004BX refers to both the 28F004BX-T and 28F004BX-B devices. The 4-Mbit flash memory family refers to both the 28F400BX and 28F004BX products. This datasheet comprises the specifications for four separate products in the 4-Mbit flash memory family. Section 1 provides an overview of the 4-Mbit flash memory family including applications, pinouts and pin descriptions. Sections 2 and 3 describe in detail the specific memory organizations for the 28F400BX and 28F004BX products respectively. Section 4 combines a description of the family's principles of operations. Finally Section 5 describes the family's operating specifications.

Product Family

X8/X16 Products	X8-Only Products
28F400BX-T	28F004BX-T
28F400BX-B	28F004BX-B

1.1 Main Features

The 28F400BX/28F004BX boot block flash memory family is a very high performance 4-Mbit (4,194,304 bit) memory family organized as either 256 KWords (262,144 words) of 16 bits each or 512 Kbytes (524,288 bytes) of 8 bits each.

Seven Separately Erasable Blocks including a **Hardware-Lockable boot block** (16,384 Bytes), **Two parameter blocks** (8,192 Bytes each) and **Four main blocks** (1 block of 98,304 Bytes and 3 blocks of 131,072 Bytes) are included on the 4-Mbit family. An erase operation erases one of the main blocks in typically 2.4 seconds and the boot or parameter blocks in typically 1.0 seconds independent of the remaining blocks. Each block can be independently erased and programmed 100,000 times.

The Boot Block is located at either the top (28F400BX-T, 28F004BX-T) or the bottom (28F400BX-B, 28F004BX-B) of the address map in order to accommodate different microprocessor protocols for boot code location. The **hardware lockable boot block** provides the most secure code storage. The boot block is intended to store the kernel code required for booting-up a system. When the **RP#** pin is between 11.4V and 12.6V the boot block is unlocked and program and erase operations can be performed. When the **RP#** pin is at or below 6.5V the boot block is locked and program and erase operations to the boot block are ignored.

The 28F400BX products are available in the ROM/EPROM compatible pinout and housed in the

44-Lead PSOP (Plastic Small Outline) package and the 56-Lead TSOP (Thin Small Outline, 1.2mm thick) package as shown in Figures 3 and 4. The 28F004BX products are available in the 40-Lead TSOP (1.2mm thick) package as shown in Figure 5.

The **Command User Interface (CUI)** serves as the interface between the microprocessor or microcontroller and the internal operation of the 28F400BX and 28F004BX flash memory products.

Program and Erase Automation allows program and erase operations to be executed using a two-write command sequence to the CUI. The internal Write State Machine (WSM) automatically executes the algorithms and timings necessary for program and erase operations, including verifications, thereby unburdening the microprocessor or microcontroller. Writing of memory data is performed in word or byte increments for the 28F400BX family and in byte increments for the 28F004BX family typically within 9 μ s which is a 100% improvement over current flash memory products.

The **Status Register (SR)** indicates the status of the WSM and whether the WSM successfully completed the desired program or erase operation.

Maximum Access Time of **60 ns (TACC)** is achieved over the commercial temperature range (0°C to 70°C), 5% V_{CC} supply voltage range (4.75V to 5.25V) and 30 pF output load. Maximum Access Time of **70 ns (TACC)** is achieved over the commercial temperature range, 10% V_{CC} supply range (4.5V to 5.5V) and 100 pF output load.

Ipp maximum Program current is 40 mA for x16 operation and 30 mA for x8 operation. Ipp Erase current is 30 mA maximum. Vpp erase and programming voltage is 11.4V to 12.6V ($V_{pp} = 12V \pm 5\%$) under all operating conditions. As an option, V_{pp} can also vary between 10.8V to 13.2V ($V_{pp} = 12V \pm 10\%$) with a guaranteed number of 100 block erase cycles.

Typical Icc Active Current of 25 mA is achieved for the X16 products (28F400BX). **Typical Icc Active Current of 20 mA** is achieved for the X8 products (28F400BX, 28F004BX). Refer to the I_{CC} active current derating curves in this datasheet.

The 4-Mbit boot block flash memory family is also designed with an Automatic Power Savings (APS) feature to minimize system battery current drain and allows for very low power designs. Once the device is accessed to read array data, APS mode will immediately put the memory in static mode of operation where I_{CC} active current is typically 1 mA until the next read is initiated.

When the CE# and RP# pins are at V_{CC} and the BYTE# pin (28F400BX-only) is at either V_{CC} or GND the **CMOS Standby** mode is enabled where I_{CC} is typically 50 μA .

A **Deep Power-Down Mode** is enabled when the RP# pin is at ground minimizing power consumption and providing write protection during power-up conditions. I_{CC} current during deep power-down mode is 0.20 μA typical. An initial maximum access time or Reset Time of 300 ns is required from RP# switching until outputs are valid. Equivalently, the device has a maximum wake-up time of 215 ns until writes to the Command User Interface are recognized. When RP# is at ground the WSM is reset, the Status Register is cleared and the entire device is protected from being written to. This feature prevents data corruption and protects the code stored in the device during system reset. The system Reset pin can be tied to RP# to reset the memory to normal read mode upon activation of the Reset pin. With on-chip program/erase automation in the 4-Mbit family and the RP# functionality for data protection, when the CPU is reset and even if a program or erase command is issued, the device will not recognize any operation until RP# returns to its normal state.

For the 28F400BX, Byte-wide or Word-wide Input/Output Control is possible by controlling the BYTE# pin. When the BYTE# pin is at a logic low the device is in the byte-wide mode (x8) and data is read and written through DQ[0:7]. During the byte-wide mode, DQ[8:14] are tri-stated and DQ15/A-1 becomes the lowest order address pin. When the BYTE# pin is at a logic high the device is in the word-wide mode (x16) and data is read and written through DQ[0:15].

1.2 Applications

The 4-Mbit boot block flash memory family combines high density, high performance, cost-effective flash memories with blocking and hardware protection capabilities. Its flexibility and versatility will reduce costs throughout the product life cycle. Flash memory is ideal for Just-In-Time production flow, reducing system inventory and costs, and eliminating component handling during the production phase.

During the product life cycle, when code updates or feature enhancements become necessary, flash memory will reduce the update costs by allowing ei-

ther a user-performed code change via floppy disk or a remote code change via a serial link. The 4-Mbit boot block flash memory family provides full function, blocked flash memories suitable for a wide range of applications. These applications include **Extended PC BIOS and ROM-able** applications storage, Digital Cellular Phone program and data storage, **Telecommunication** boot/firmware, **Printer** firmware/font storage and various other embedded applications where both program and data storage are required.

Reprogrammable systems such as personal computers, are ideal applications for the 4-Mbit flash memory products. Portable and handheld personal computer applications are becoming more complex with the addition of power management software to take advantage of the latest microprocessor technology, the availability of ROM-based application software, pen tablet code for electronic hand writing, and diagnostic code. Figure 1 shows an example of a 28F400BX-T application.

This increase in software sophistication augments the probability that a code update will be required after the PC is shipped. The 4-Mbit flash memory products provide an inexpensive update solution for the notebook and handheld personal computers while extending their product lifetime. Furthermore, the 4-Mbit flash memory products' power-down mode provides added flexibility for these battery-operated portable designs which require operation at very low power levels.

The 4-Mbit flash memory products also provide excellent design solutions for Digital Cellular Phone and Telecommunication switching applications requiring high performance, high density storage capability coupled with modular software designs, and a small form factor package (X3-only bus). The 4-Mbit's blocking scheme allows for an easy segmentation of the embedded code with; 16 Kbytes of Hardware-Protected Boot code, 4 Main Blocks of program code and 2 Parameter Blocks of 8 Kbytes each for frequently updatable data storage and diagnostic messages (e.g., phone numbers, authorization codes). Figure 2 is an example of such an application with the 28F004BX-T.

These are a few actual examples of the wide range of applications for the 4-Mbit Boot Block flash memory family which enable system designers achieve the best possible product design. Only your imagination limits the applicability of such a versatile product family.



The 28F400BX 44-Lead PSOP pinout follows the industry standard ROM/EPROM pinout as shown in Figure 3. Furthermore, the 28F400BX 56-Lead TSOP pinout shown in Figure 4 provides density upgrades to future higher density boot block memories.

ure 5 is 100% compatible and provides a density upgrade for the 2-Mbit Boot Block flash memory or the 28F002BX.

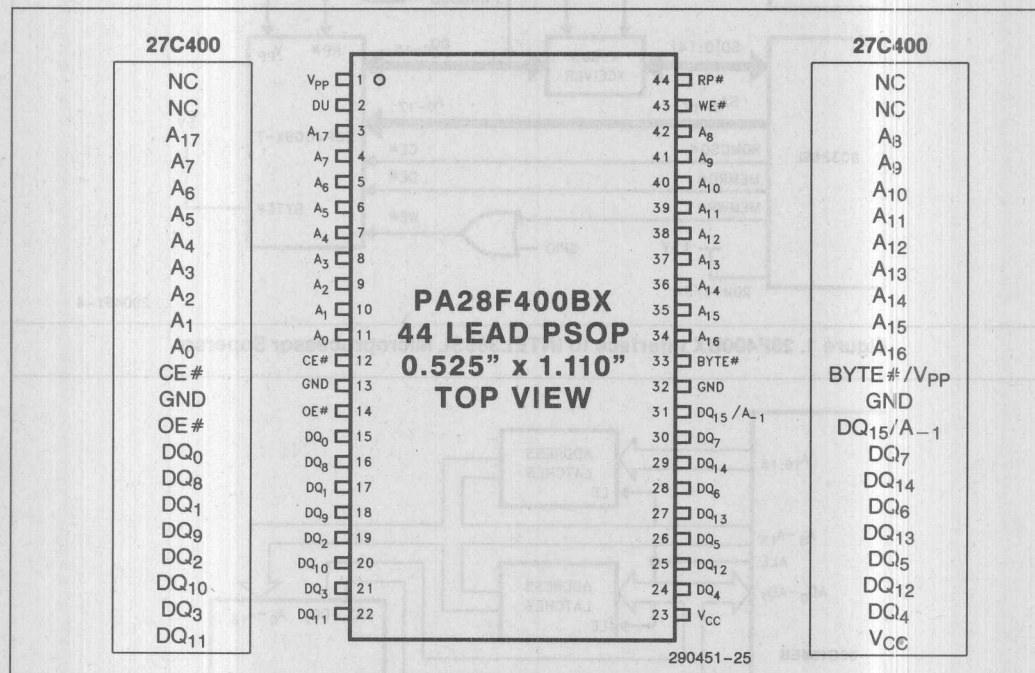


Figure 3. PSOP Lead Configuration for x8/x16 28F400BX

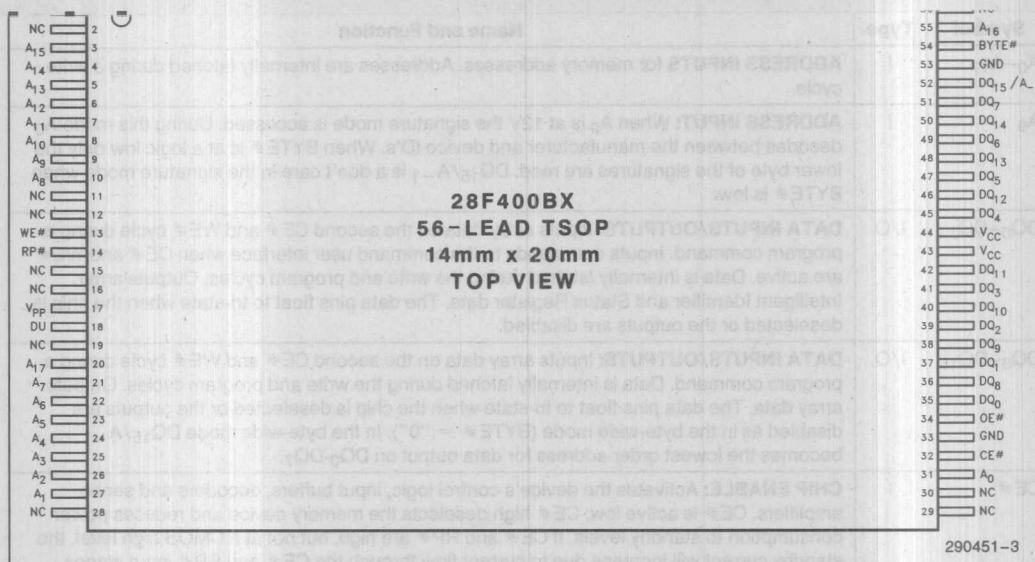


Figure 4. TSOP Lead Configuration for x8/x16 28F400BX

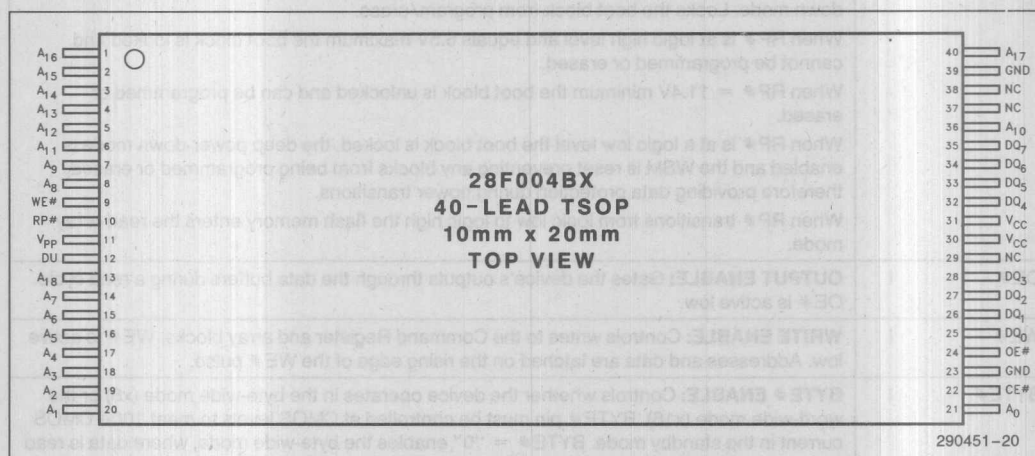


Figure 5. TSOP Lead Configuration for x8 28F004BX

1.4 28F400BX Pin Descriptions

Symbol	Type	Name and Function
A ₀ –A ₁₇	I	ADDRESS INPUTS for memory addresses. Addresses are internally latched during a write cycle.
A ₉	I	ADDRESS INPUT: When A ₉ is at 12V the signature mode is accessed. During this mode A ₀ decodes between the manufacturer and device ID's. When BYTE# is at a logic low only the lower byte of the signatures are read. DQ ₁₅ /A _{–1} is a don't care in the signature mode when BYTE# is low.
DQ ₀ –DQ ₇	I/O	DATA INPUTS/OUTPUTS: Inputs array data on the second CE# and WE# cycle during a program command. Inputs commands to the command user interface when CE# and WE# are active. Data is internally latched during the write and program cycles. Outputs array, Intelligent Identifier and Status Register data. The data pins float to tri-state when the chip is deselected or the outputs are disabled.
DQ ₈ –DQ ₁₅	I/O	DATA INPUTS/OUTPUTS: Inputs array data on the second CE# and WE# cycle during a program command. Data is internally latched during the write and program cycles. Outputs array data. The data pins float to tri-state when the chip is deselected or the outputs are disabled as in the byte-wide mode (BYTE# = "0"). In the byte-wide mode DQ ₁₅ /A _{–1} becomes the lowest order address for data output on DQ ₀ –DQ ₇ .
CE#	I	CHIP ENABLE: Activates the device's control logic, input buffers, decoders and sense amplifiers. CE# is active low; CE# high deselects the memory device and reduces power consumption to standby levels. If CE# and RP# are high, but not at a CMOS high level, the standby current will increase due to current flow through the CE# and RP# input stages.
RP#	I	RESET/DEEP POWER-DOWN: Provides three-state control. Puts the device in deep power-down mode. Locks the boot block from program/erase. When RP# is at logic high level and equals 6.5V maximum the boot block is locked and cannot be programmed or erased. When RP# = 11.4V minimum the boot block is unlocked and can be programmed or erased. When RP# is at a logic low level the boot block is locked, the deep power-down mode is enabled and the WSM is reset preventing any blocks from being programmed or erased, therefore providing data protection during power transitions. When RP# transitions from logic low to logic high the flash memory enters the read-array mode.
OE#	I	OUTPUT ENABLE: Gates the device's outputs through the data buffers during a read cycle. OE# is active low.
WE#	I	WRITE ENABLE: Controls writes to the Command Register and array blocks. WE# is active low. Addresses and data are latched on the rising edge of the WE# pulse.
BYTE#	I	BYTE# ENABLE: Controls whether the device operates in the byte-wide mode (x8) or the word-wide mode (x16). BYTE# pin must be controlled at CMOS levels to meet 100A CMOS current in the standby mode. BYTE# = "0" enables the byte-wide mode, where data is read and programmed on DQ ₀ –DQ ₇ and DQ ₁₅ /A _{–1} becomes the lowest order address that decodes between the upper and lower byte. DQ ₈ –DQ ₁₄ are tri-stated during the byte-wide mode. BYTE# = "1" enables the word-wide mode where data is read and programmed on DQ ₀ –DQ ₁₅ .
V _{PP}		PROGRAM/ERASE POWER SUPPLY: For erasing memory array blocks or programming data in each block. Note: V _{PP} < V _{PPLMAX} memory contents cannot be altered.
V _{CC}		DEVICE POWER SUPPLY (5V ± 10%, 5V ± 5%)
GND		GROUND: For all internal circuitry.
NC		NO CONNECT: Pin may be driven or left floating.
DU		DON'T USE PIN: Pin should not be connected to anything.

1.5 28F004BX Pin Descriptions

Symbol	Type	Name and Function
A ₀ –A ₁₈	I	ADDRESS INPUTS for memory addresses. Addresses are internally latched during a write cycle.
A ₉	I	ADDRESS INPUT: When A ₉ is at 12V the signature mode is accessed. During this mode A ₀ decodes between the manufacturer and device ID's.
DQ ₀ –DQ ₇	I/O	DATA INPUTS/OUTPUTS: Inputs array data on the second CE # and WE # cycle during a program command. Inputs commands to the command user interface when CE # and WE # are active. Data is internally latched during the write and program cycles. Outputs array, Intelligent Identifier and status register data. The data pins float to tri-state when the chip is deselected or the outputs are disabled.
CE #	I	CHIP ENABLE: Activates the device's control logic, input buffers, decoders and sense amplifiers. CE # is active low; CE # high deselects the memory device and reduces power consumption to standby levels. If CE # and RP # are high, but not at a CMOS high level, the standby current will increase due to current flow through the CE # and RP # input stages.
RP #	I	RESET/DEEP POWERDOWN: Provides Three-State control. Puts the device in deep power-down mode. Locks the Boot Block from program/erase. When RP # is at logic high level and equals 6.5V maximum the Boot Block is locked and cannot be programmed or erased. When RP # = 11.4V minimum the Boot Block is unlocked and can be programmed or erased. When RP # is at a logic low level the Boot Block is locked, the deep power-down mode is enabled and the WSM is reset preventing any blocks from being programmed or erased, therefore providing data protection during power transitions. When RP # transitions from logic low to logic high, the flash memory enters the read-array mode.
OE #	I	OUTPUT ENABLE: Gates the device's outputs through the data buffers during a read cycle. OE # is active low.
WE #	I	WRITE ENABLE: Controls writes to the Command Register and array blocks. WE # is active low. Addresses and data are latched on the rising edge of the WE # pulse.
V _{PP}		PROGRAM/ERASE POWER SUPPLY: For erasing memory array blocks or programming data in each block. NOTE: V _{PP} < V _{PPLMAX} memory contents cannot be altered.
V _{CC}		DEVICE POWER SUPPLY (5V ± 10%, 5V ± 5%)
GND		GROUND: For all internal circuitry.
NC		NO CONNECT: Pin may be driven or left floating.
DU		DON'T USE PIN: Pin should not be connected to anything.

2.0 28F400BX WORD/BYTE-WIDE PRODUCTS DESCRIPTION

290451-1

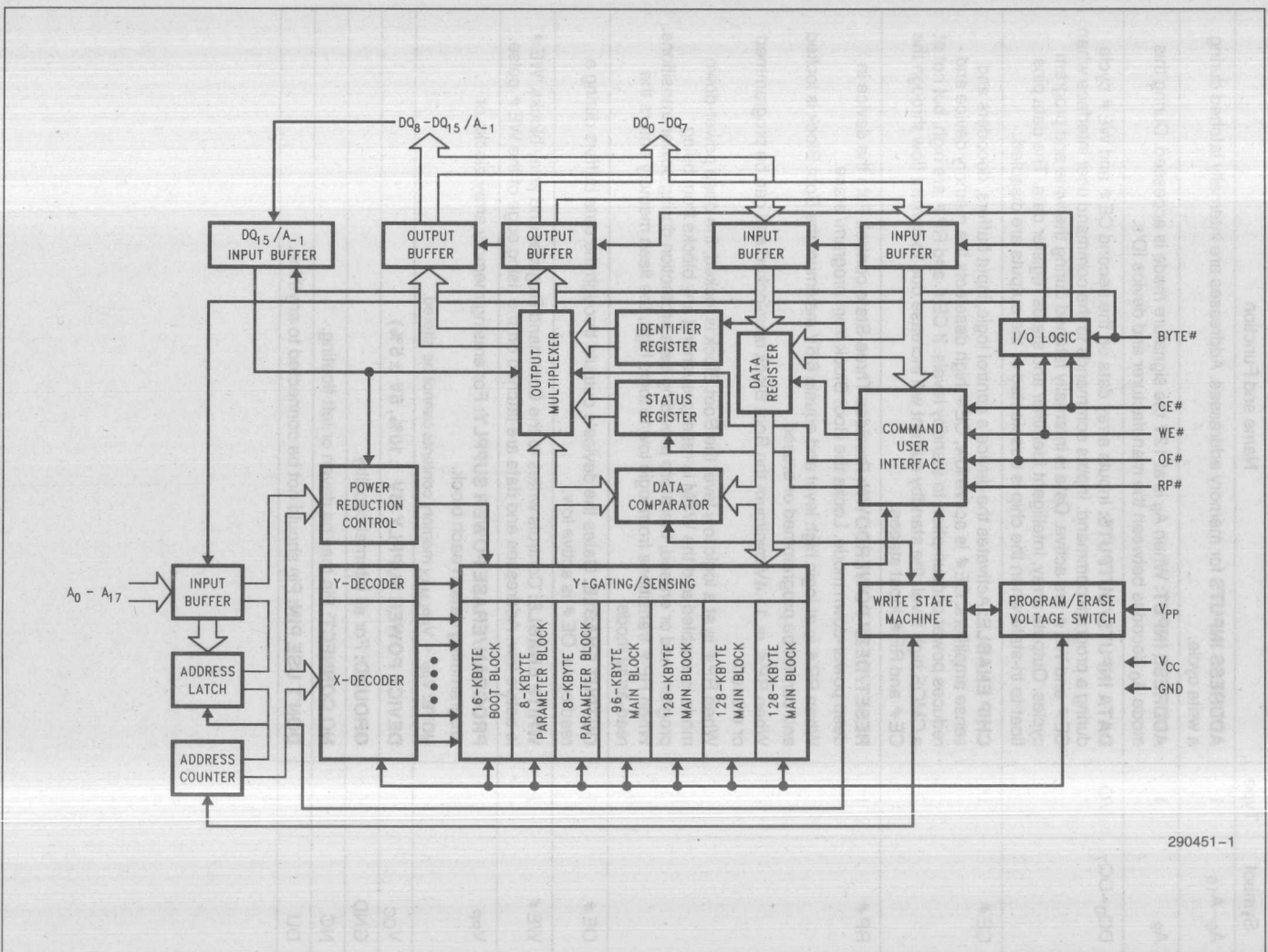


Figure 6. 28F400BX Word/Byte Block Diagram

2.1 28F400BX Memory Organization

2.1.1 BLOCKING

The 28F400BX uses a blocked array architecture to provide independent erasure of memory blocks. A block is erased independently of other blocks in the array when an address is given within the block address range and the Erase Setup and Erase Confirm commands are written to the CUI. The 28F400BX is a random read/write memory, only erasure is performed by block.

2.1.1.1 Boot Block Operation and Data Protection

The 16-Kbyte boot block provides a lock feature for secure code storage. The intent of the boot block is to provide a secure storage area for the kernel code that is required to boot a system in the event of power failure or other disruption during code update. This lock feature ensures absolute data integrity by preventing the boot block from being written or erased when RP# is not at 12V. The boot block can be erased and written when RP# is held at 12V for the duration of the erase or program operation. This allows customers to change the boot code when necessary while providing security when needed. See the Block Memory Map section for address locations of the boot block for the 28F400BX-T and 28F400BX-B.

2.1.1.2 Parameter Block Operation

The 28F400BX has 2 parameter blocks (8-Kbytes each). The parameter blocks are intended to provide storage for frequently updated system parameters and configuration or diagnostic information. The parameter blocks can also be used to store additional boot or main code. The parameter blocks however, do not have the hardware write protection feature that the boot block has. The parameter blocks provide for more efficient memory utilization when dealing with parameter changes versus regularly blocked devices. See the Block Memory Map section for address locations of the parameter blocks for the 28F400BX-T and 28F400BX-B.

2.1.1.3 Main Block Operation

Four main blocks of memory exist on the 28F400BX (3 x 128-Kbyte blocks and 1 x 96-Kbyte blocks). See the following section on Block Memory Map for the address location of these blocks for the 28F400BX-T and 28F400BX-B products.

2.1.2 BLOCK MEMORY MAP

Two versions of the 28F400BX product exist to support two different memory maps of the array blocks in order to accommodate different microprocessor protocols for boot code location. The 28F400BX-T memory map is inverted from the 28F400BX-B memory map.

2.1.2.1. 28F400BX-B Memory Map

The 28F400BX-B device has the 16-Kbyte boot block located from 00000H to 01FFFFH to accommodate those microprocessors that boot from the bottom of the address map at 00000H. In the 28F400BX-B the first 8-Kbyte parameter block resides in memory space from 02000H to 02FFFFH. The second 8-Kbyte parameter block resides in memory space from 03000H to 03FFFFH. The 96-Kbyte main block resides in memory space from 04000H to 0FFFFH. The three 128-Kbyte main block resides in memory space from 10000H to 1FFFFH, 20000H to 2FFFFH and 30000H to 3FFFFH (word locations). See Figure 7.

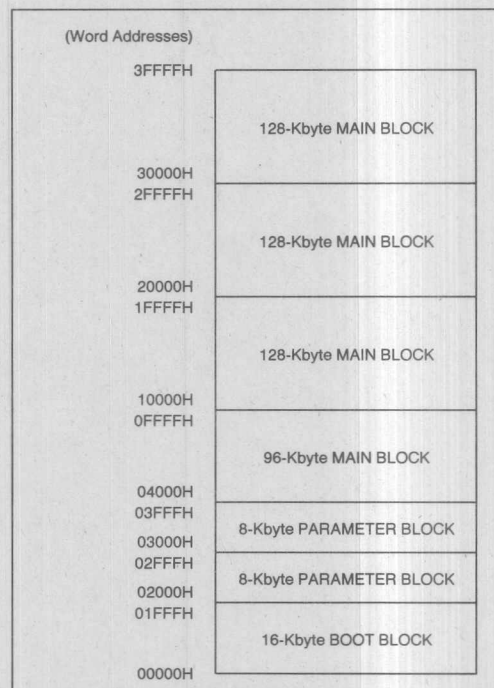


Figure 7. 28F400BX-B Memory Map

2.1.2.2 28F400BX-T Memory Map

The 28F400BX-T device has the 16-Kbyte boot block located from 3E000H to 3FFFFH to accommodate those microprocessors that boot from the top of the address map. In the 28F400BX-T the first 8-Kbyte parameter block resides in memory space from 3D000H to 3DFFFH. The second 8-Kbyte parameter block resides in memory space from 3C000H to 3CFFFH. The 96-Kbyte main block resides in memory space from 30000H to 3BFFFH. The three 128-Kbyte main blocks reside in memory space from 20000H to 2FFFFH, 10000H to 1FFFFH and 00000H to 0FFFFH as shown below in Figure 8.

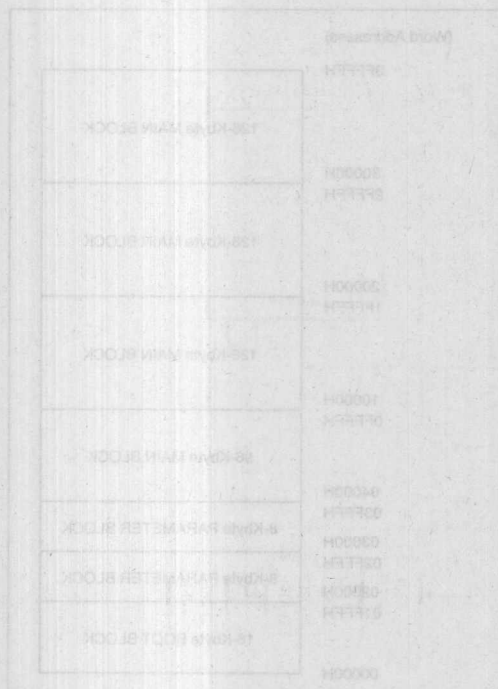


Figure 7. 28F400BX-B Memory Map

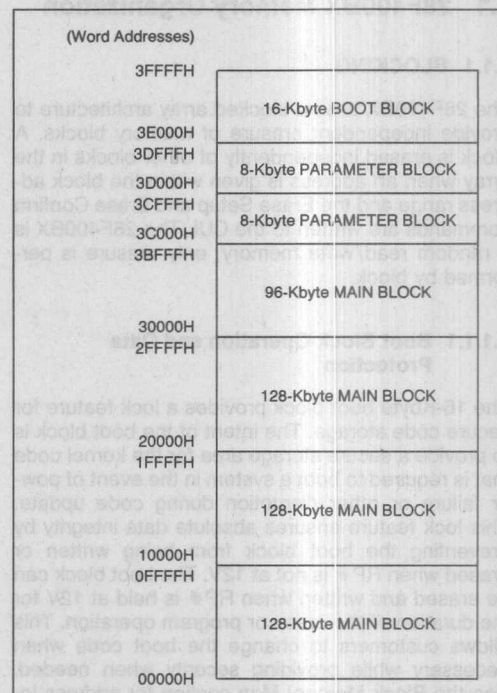


Figure 8. 28F400BX-T Memory Map

290451-19



Figure 9. 28F004BX Byte-Wide Block Diagram

3.1 28F004BX Memory Organization

3.1.1 BLOCKING

The 28F004BX uses a blocked array architecture to provide independent erasure of memory blocks. A block is erased independently of other blocks in the array when an address is given within the block address range and the Erase Setup and Erase Confirm commands are written to the CUI. The 28F004BX is a random read/write memory, only erasure is performed by block.

3.1.1.1 Boot Block Operation and Data Protection

The 16-Kbyte boot block provides a lock feature for secure code storage. The intent of the boot block is to provide a secure storage area for the kernel code that is required to boot a system in the event of power failure or other disruption during code update. This lock feature ensures absolute data integrity by preventing the boot block from being programmed or erased when RP# is not at 12V. The boot block can be erased and programmed when RP# is held at 12V for the duration of the erase or program operation. This allows customers to change the boot code when necessary while still providing security when needed. See the Block Memory Map section for address locations of the boot block for the 28F004BX-T and 28F004BX-B.

3.1.1.2 Parameter Block Operation

The 28F004BX has 2 parameter blocks (8-Kbytes each). The parameter blocks are intended to provide storage for frequently updated system parameters and configuration or diagnostic information. The parameter blocks can also be used to store additional boot or main code. The parameter blocks however, do not have the hardware write protection feature that the boot block has. Parameter blocks provide for more efficient memory utilization when dealing with small parameter changes versus regularly blocked devices. See the Block Memory Map section for address locations of the parameter blocks for the 28F004BX-T and 28F004BX-B.

3.1.1.3 Main Block Operation

Four main blocks of memory exist on the 28F004BX (3 × 128-Kbyte blocks and 1 × 96-Kbyte blocks). See the following section on Block Memory Map for the address location of these blocks for the 28F004BX-T and 28F004BX-B.

3.1.2 BLOCK MEMORY MAP

Two versions of the 28F004BX product exist to support two different memory maps of the array blocks in order to accommodate different microprocessor protocols for boot code location. The 28F004BX-T memory map is inverted from the 28F004BX-B memory map.

3.1.2.1 28F004BX-B Memory Map

The 28F004BX-B device has the 16-Kbyte boot block located from 00000H to 03FFFFH to accommodate those microprocessors that boot from the bottom of the address map at 00000H. In the 28F004BX-B the first 8-Kbyte parameter block resides in memory from 04000H to 05FFFFH. The second 8-Kbyte parameter block resides in memory space from 06000H to 07FFFFH. The 96-Kbyte main block resides in memory space from 08000H to 1FFFFH. The three 128-Kbyte main block reside in memory space from 20000H to 3FFFFH, 40000H to 5FFFFH and 60000H to 7FFFFH. See Figure 10.

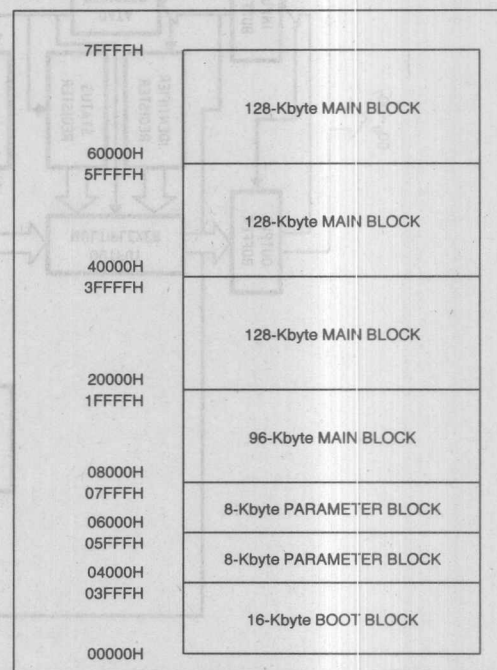


Figure 10. 28F004BX-B Memory Map

The 28F004BX-T device has the 16-Kbyte boot block located from 7C000H to 7FFFFH to accommodate those microprocessors that boot from the top of the address map. In the 28F004BX-T the first 8-Kbyte parameter block resides in memory space from 7A000H to 7BFFFFH. The second 8-Kbyte parameter block resides in memory space from 78000H to 79FFFFH. The 96-Kbyte main block resides in memory space from 60000H to 77FFFFH. The three 128-Kbyte main blocks reside in memory space from 40000H to 5FFFFH, 20000H to 3FFFFH and 00000H to 1FFFFH.

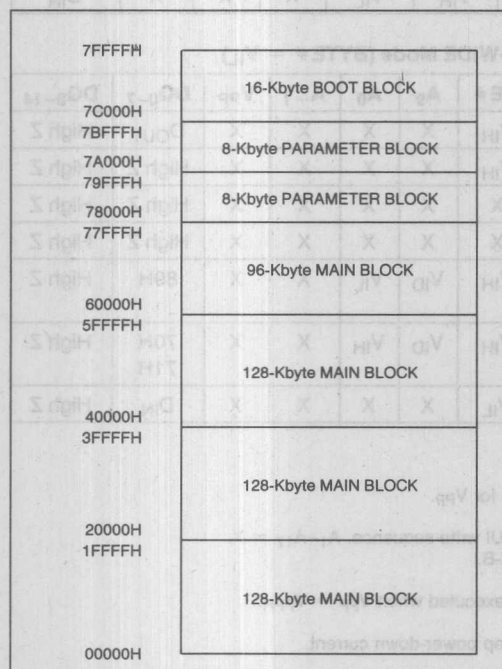


Figure 11. 28F004BX-T Memory Map

OF OPERATION

Flash memory augments EPROM functionality with in-circuit electrical write and erase. The 4-Mbit flash family utilizes a Command User Interface (CUI) and internally generated and timed algorithms to simplify write and erase operations.

The CUI allows for 100% TTL-level control inputs, fixed power supplies during erasure and programming, and maximum EPROM compatibility.

In the absence of high voltage on the V_{PP} pin, the 4-Mbit boot block flash family will only successfully execute the following commands: Read Array, Read Status Register, Clear Status Register and Intelligent Identifier mode. The device provides standard EPROM read, standby and output disable operations. Manufacturer Identification and Device Identification data can be accessed through the CUI or through the standard EPROM A₉ high voltage access (V_{ID}) for PROM programming equipment.

The same EPROM read, standby and output disable functions are available when high voltage is applied to the V_{PP} pin. In addition, high voltage on V_{PP} allows write and erase of the device. All functions associated with altering memory contents: write and erase, Intelligent Identifier read and Read Status are accessed via the CUI.

The purpose of the Write State Machine (WSM) is to completely automate the write and erasure of the device. The WSM will begin operation upon receipt of a signal from the CUI and will report status back through a Status Register. The CUI will handle the WE# interface to the data and address latches, as well as system software requests for status while the WSM is in operation.

4.1 28F400BX Bus Operations

Flash memory reads, erases and writes in-system via the local CPU. All bus cycles to or from the flash memory conform to standard microprocessor bus cycles.

Table 1. Bus Operations for WORD-WIDE Mode (BYTE# = V_{IH})

Mode	Notes	RP#	CE#	OE#	WE#	A ₉	A ₀	V _{PP}	DQ ₀₋₁₅
Read	1, 2, 3	V _{IH}	V _{IL}	V _{IL}	V _{IH}	X	X	X	D _{OUT}
Output Disable		V _{IH}	V _{IL}	V _{IH}	V _{IH}	X	X	X	High Z
Standby		V _{IH}	V _{IH}	X	X	X	X	X	High Z
Deep Power-Down	9	V _{IL}	X	X	X	X	X	X	High Z
Intelligent Identifier (Mfr)	4	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IL}	X	0089H
Intelligent Identifier (Device)	4, 5	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IH}	X	4470H 4471H
Write	6, 7, 8	V _{IH}	V _{IL}	V _{IH}	V _{IL}	X	X	X	D _{IN}

Table 2. Bus Operations for BYTE-WIDE Mode (BYTE# = V_{IL})

Mode	Notes	RP#	CE#	OE#	WE#	A ₉	A ₀	A ₋₁	V _{PP}	DQ ₀₋₇	DQ ₈₋₁₄
Read	1, 2, 3	V _{IH}	V _{IL}	V _{IL}	V _{IH}	X	X	X	X	D _{OUT}	High Z
Output Disable		V _{IH}	V _{IL}	V _{IH}	V _{IH}	X	X	X	X	High Z	High Z
Standby		V _{IH}	V _{IH}	X	X	X	X	X	X	High Z	High Z
Deep Power-Down	9	V _{IL}	X	X	X	X	X	X	X	High Z	High Z
Intelligent Identifier (Mfr)	4	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IL}	X	X	89H	High Z
Intelligent Identifier (Device)	4, 5	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IH}	X	X	70H 71H	High Z
Write	6, 7, 8	V _{IH}	V _{IL}	V _{IH}	V _{IL}	X	X	X	X	D _{IN}	High Z

NOTES:

1. Refer to DC Characteristics.
2. X can be V_L, V_{IH} for control pins and addresses, V_{PL} or V_{PPH} for V_{PP}.
3. See DC Characteristics for V_{PL}, V_{PPH}, V_{HH}, V_{ID} voltages.
4. Manufacturer and Device codes may also be accessed via a CUI write sequence. A₁-A₁₇ = X.
5. Device ID = 4470H for 28F400BX-T and 4471H for 28F400BX-B.
6. Refer to Table 4 for valid D_{IN} during a write operation.
7. Command writes for Block Erase or Word/Byte Write are only executed when V_{PP} = V_{PPH}.
8. To write or erase the boot block, hold RP# at V_{HH}.
9. RP# must be at GND ±0.2V to meet the 1.2 μA maximum deep power-down current.

4.2 28F004BX Bus Operations

Table 3. Bus Operations

Mode	Notes	RP #	CE #	OE #	WE #	A ₉	A ₀	V _{PP}	DQ ₀₋₇
Read	1, 2, 3	V _{IH}	V _{IL}	V _{IL}	V _{IH}	X	X	X	D _{OUT}
Output Disable		V _{IH}	V _{IL}	V _{IH}	V _{IH}	X	X	X	High Z
Standby		V _{IH}	V _{IH}	X	X	X	X	X	High Z
Deep Power-Down	9	V _{IL}	X	X	X	X	X	X	High Z
Intelligent Identifier (Mfr)	4	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IL}	X	89H
Intelligent Identifier (Device)	4, 5	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IH}	X	78H 79H
Write	6, 7, 8	V _{IH}	V _{IL}	V _{IH}	V _{IL}	X	X	X	D _{IN}

NOTES:

1. Refer to DC Characteristics.
2. X can be V_{IL} or V_{IH} for control pins and addresses, V_{PPL} or V_{PPH} for V_{PP}.
3. See DC Characteristics for V_{PPL}, V_{PPH}, V_{IH}, V_{ID} voltages.
4. Manufacturer and Device codes may also be accessed via a CUI write sequence. A₁-A₁₈ = X.
5. Device ID = 78H for 28F004BX-T and 79H for 28F004BX-B.
6. Refer to Table 4 for valid D_{IN} during a write operation.
7. Command writes for Block erase or byte program are only executed when V_{PP} = V_{PPH}.
8. Program or erase the Boot block by holding RP# at V_{IH}.
9. RP# must be at GND ±0.2V to meet the 1.2 µA maximum deep power-down current.

4.3 Read Operations

The 4-Mbit boot block flash family has three user read modes; Array, Intelligent Identifier, and Status Register. Status Register read mode will be discussed in detail in the "Write Operations" section.

During power-up conditions (V_{CC} supply ramping), it takes a maximum of 600 ns from when V_{CC} is at 4.5V minimum to valid data on the outputs.

4.3.1 READ ARRAY

If the memory is not in the Read Array mode, it is necessary to write the appropriate read mode command to the CUI. The 4-Mbit boot block flash family has three control functions, all of which must be logically active, to obtain data at the outputs. Chip-Enable CE# is the device selection control. Power-Down RP# is the device power control. Output-Enable OE# is the DATA INPUT/OUTPUT (DQ[0:15] or DQ[0:7]) direction control and when active is used to drive data from the selected memory on to the I/O bus.

4.3.1.1 Output Control

With OE# at logic-high level (V_{IH}), the output from the device is disabled and data input/output pins (DQ[0:15] or DQ[0:7]) are tri-stated. Data input is then controlled by WE#.

4.3.1.2 Input Control

With WE# at logic-high level (V_{IH}), input to the device is disabled. Data Input/Output pins (DQ[0:15] or DQ[0:7]) are controlled by OE#.

4.3.2 INTELLIGENT IDENTIFIERS

28F400BX PRODUCTS

The manufacturer and device codes are read via the CUI or by taking the A₉ pin to 12V. Writing 90H to the CUI places the device into Intelligent Identifier read mode. A read of location 00000H outputs the manufacturer's identification code, 0089H, and location 00001H outputs the device code; 4470H for 28F400BX-T, 4471H for 28F400BX-B. When BYTE# is at a logic low only the lower byte of the above signatures is read and DQ₁₅/A₋₁ is a "don't care" during Intelligent Identifier mode. A read array command must be written to the memory to return to the read array mode.

28F004BX PRODUCTS

The manufacturer and device codes are also read via the CUI or by taking the A₉ pin to 12V. Writing 90H to the CUI places the device into Intelligent Identifier read mode. A read of location 00000H outputs the manufacturer's identification code, 89H, and location 00001H outputs the device code; 78H for 28F004BX-T, 79H for 28F004BX-B.

4.4 Write Operations

Commands are written to the CUI using standard microprocessor write timings. The CUI serves as the interface between the microprocessor and the internal chip operation. The CUI can decipher Read Array, Read Intelligent Identifier, Read Status Register, Clear Status Register, Erase and Program commands. In the event of a read command, the CUI simply points the read path at either the array, the Intelligent Identifier, or the status register depending on the specific read command given. For a program or erase cycle, the CUI informs the write state machine that a write or erase has been requested. During a program cycle, the Write State Machine will control the program sequences and the CUI will only respond to status reads. During an erase cycle, the CUI will respond to status reads and erase suspend. After the Write State Machine has completed its task, it will allow the CUI to respond to its full command set. The CUI will stay in the current command state until the microprocessor issues another command.

The CUI will successfully initiate an erase or write operation only when V_{pp} is within its voltage range. Depending upon the application, the system designer may choose to make the V_{pp} power supply switchable, available only when memory updates are desired. The system designer can also choose to "hard-wire" V_{pp} to 12V. The 4-Mbit boot block flash family is designed to accommodate—either design practice. It is recommended that $RP\#$ be tied to logical Reset for data protection during unstable CPU reset function as described in the "Product Family Overview" section.

4.4.1 BOOT BLOCK WRITE OPERATIONS

In the case of Boot Block modifications (write and erase), $RP\#$ is set to $V_{HH} = 12V$ typically, in addition to V_{pp} at high voltage.

However, if $RP\#$ is not at V_{HH} when a program or erase operation of the boot block is attempted, the corresponding status register bit (Bit 4 for Program and Bit 5 for Erase, refer to Table 5 for Status Register Definitions) is set to indicate the failure to complete the operation.

4.4.2 COMMAND USER INTERFACE (CUI)

The Command User Interface (CUI) serves as the interface to the microprocessor. The CUI points the read/write path to the appropriate circuit block as described in the previous section. After the WSM has completed its task, it will set the WSM Status bit to a "1", which will also allow the CUI to respond to its full command set. Note that after the WSM has returned control to the CUI, the CUI will remain in its current state.

4.4.2.1 Command Set

Command Codes	Device Mode
00	Invalid/Reserved
10	Alternate Program Setup
20	Erase Setup
40	Program Setup
50	Clear Status Register
70	Read Status Register
90	Intelligent Identifier
B0	Erase Suspend
D0	Erase Resume/Erase Confirm
FF	Read Array

4.4.2.2 Command Function Descriptions

Device operations are selected by writing specific commands into the CUI. Table 4 defines the 4-Mbit boot block flash family commands.

Table 4. Command Definitions

Command	Bus Cycles Req'd	Notes	First Bus Cycle			Second Bus Cycle		
			Operation	Address	Data	Operation	Address	Data
Read Array	1	1	Write	X	FFH			
Intelligent Identifier	3	2, 4	Write	X	90H	Read	IA	IID
Read Status Register	2	3	Write	X	70H	Read	X	SRD
Clear Status Register	1		Write	X	50H			
Erase Setup/Erase Confirm	2	5	Write	BA	20H	Write	BA	D0H
Word/Byte Write Setup/Write	2	6, 7	Write	WA	40H	Write	WA	WD
Erase Suspend/Erase Resume	2		Write	X	B0H	Write	X	D0H
Alternate Word/Byte Write Setup/Write	2	6, 7	Write	WA	10H	Write	WA	WD

NOTES:

1. Bus operations are defined in Tables 1, 2, 3.
 2. IA = Identifier Address: 00H for manufacturer code, 01H for device code.
 3. SRD = Data read from Status Register.
 4. IID = Intelligent Identifier Data.
- Following the Intelligent Identifier Command, two read operations access manufacturer and device codes.
5. BA = Address within the block being erased.
 6. WA = Address to be written.
 - WD = Data to be written at location WD.
 7. Either 40H or 10H commands is valid.
 8. When writing commands to the device, the upper data bus [DQ₈-DQ₁₅] = X (28F400BX-only) which is either V_{CC} or V_{SS} to avoid burning additional current.

Invalid/Reserved

These are unassigned commands. It is not recommended that the customer use any command other than the valid commands specified above. Intel reserves the right to redefine these codes for future functions.

Read Array (FFH)

This single write command points the read path at the array. If the host CPU performs a CE#/OE# controlled read immediately following a two-write sequence that started the WSM, then the device will output status register contents. If the Read Array command is given after Erase Setup the device is reset to read the array. A two Read Array command sequence (FFH) is required to reset to Read Array after Program Setup.

Intelligent Identifier (90H)

After this command is executed, the CUI points the output path to the Intelligent Identifier circuits. Only Intelligent Identifier values at addresses 0 and 1 can be read (only address A₀ is used in this mode, all other address inputs are ignored).

Read Status Register (70H)

This is one of the two commands that is executable while the state machine is operating. After this command is written, a read of the device will output the contents of the status register, regardless of the address presented to the device.

The device automatically enters this mode after program or erase has completed.

Clear Status Register (50H)

The WSM can only set the Program Status and Erase Status bits in the status register, it can not clear them. Two reasons exist for operating the status register in this fashion. The first is a synchronization. The WSM does not know when the host CPU has read the status register, therefore it would not know when to clear the status bits. Secondly, if the CPU is programming a string of bytes, it may be more efficient to query the status register after programming the string. Thus, if any errors exist while programming the string, the status register will return the accumulated error status.

Program Setup (40H or 10H)

This command simply sets the CUI into a state such that the next write will load the address and data registers. Either 40H or 10H can be used for Program Setup. Both commands are included to accommodate efforts to achieve an industry standard command code set.

Program

The second write after the program setup command, will latch addresses and data. Also, the CUI initiates the WSM to begin execution of the program algorithm. While the WSM finishes the algorithm, the device will output Status Register contents. Note that the WSM cannot be suspended during programming.

Erase Setup (20H)

Prepares the CUI for the Erase Confirm command. No other action is taken. If the next command is not an Erase Confirm command then the CUI will set both the Program Status and Erase Status bits of the Status Register to a "1", place the device into the Read Status Register state, and wait for another command.

Erase Confirm (D0H)

If the previous command was an Erase Setup command, then the CUI will enable the WSM to erase, at the same time closing the address and data latches, and respond only to the Read Status Register and Erase Suspend commands. While the WSM is executing, the device will output Status Register data when OE# is toggled low. Status Register data can only be updated by toggling either OE# or CE# low.

Erase Suspend (B0H)

This command only has meaning while the WSM is executing an Erase operation, and therefore will only be responded to during an erase operation. After this command has been executed, the CUI will set an output that directs the WSM to suspend Erase operations, and then return to responding to only Read Status Register or to the Erase Resume commands. Once the WSM has reached the Suspend state, it will set an output into the CUI which allows the CUI to respond to the Read Array, Read Status Register, and Erase Resume commands. In this mode, the CUI will not respond to any other commands. The WSM will also set the WSM Status bit to a "1". The WSM will continue to run, idling in the SUSPEND state, regardless of the state of all input

control pins, with the exclusion of RP#. RP# will immediately shut down the WSM and the remainder of the chip. During a suspend operation, the data and address latches will remain closed, but the address pads are able to drive the address into the read path.

Erase Resume (D0H)

This command will cause the CUI to clear the Suspend state and set the WSM Status bit to a "0", but only if an Erase Suspend command was previously issued. Erase Resume will not have any effect in all other conditions.

4.4.3 STATUS REGISTER

The 4-Mbit boot block flash family contains a status register which may be read to determine when a program or erase operation is complete, and whether that operation completed successfully. The status register may be read at any time by writing the Read Status command to the CUI. After writing this command, all subsequent Read operations output data from the status register until another command is written to the CUI. A Read Array command must be written to the CUI to return to the Read Array mode.

The status register bits are output on DQ[0:7] whether the device is in the byte-wide (x8) or word-wide (x16) mode for the 28F400BX. In the word-wide mode the upper byte, DQ[8:15] is set to 00H during a Read Status command. In the byte-wide mode, DQ[8:14] are tri-stated and DQ_{15/A-1} retains the low order address function.

It should be noted that the contents of the status register are latched on the falling edge of OE# or CE# whichever occurs last in the read cycle. This prevents possible bus errors which might occur if the contents of the status register change while reading the status register. CE# or OE# must be toggled with each subsequent status read, or the completion of a program or erase operation will not be evident.

The Status Register is the interface between the microprocessor and the Write State Machine (WSM). When the WSM is active, this register will indicate the status of the WSM, and will also hold the bits indicating whether or not the WSM was successful in performing the desired operation. The WSM sets status bits "Three" through "Seven" and clears bits "Six" and "Seven", but cannot clear status bits "Three" through "Five". These bits can only be cleared by the controlling CPU through the use of the Clear Status Register command.

4.4.3.1 Status Register Bit Definition

Table 5. Status Register Definitions

WSMS	ESS	ES	PS	VPPS	R	R	R
7	6	5	4	3	2	1	0
<p>SR.7 = WRITE STATE MACHINE STATUS 1 = Ready 0 = Busy</p> <p>SR.6 = ERASE SUSPEND STATUS 1 = Erase Suspended 0 = Erase in Progress/Completed</p> <p>SR.5 = ERASE STATUS 1 = Error in Block Erasure 0 = Successful Block Erase</p> <p>SR.4 = PROGRAM STATUS 1 = Error In Byte/Word Program 0 = Successful Byte/Word Program</p> <p>SR.3 = V_{pp} STATUS 1 = V_{pp} Low Detect; Operation Abort 0 = V_{pp} OK</p> <p>SR.2-SR.0 = RESERVED FOR FUTURE ENHANCEMENTS</p>							
<p>NOTES:</p> <p>Write State Machine Status bit must first be checked to determine byte/word program or block erase completion, before the Program or Erase Status bits are checked for success.</p> <p>When Erase Suspend is issued, WSM halts execution and sets both WSMS and ESS bits to "1". ESS bit remains set to "1" until an Erase Resume command is issued.</p> <p>When this bit is set to "1". WSM has applied the maximum number of erase pulses to the block and is still unable to successfully perform an erase verify.</p> <p>When this bit is set to "1", WSM has attempted but failed to Program a byte or word.</p> <p>The V_{pp} Status bit unlike an A/D converter, does not provide continuous indication of V_{pp} level. The WSM interrogates the V_{pp} level only after the byte write or block erase command sequences have been entered and informs the system if V_{pp} has not been switched on. The V_{pp} Status bit is not guaranteed to report accurate feedback between V_{ppL} and V_{ppH}.</p> <p>These bits are reserved for future use and should be masked out when polling the Status Register.</p>							

4.4.3.2 Clearing the Status Register

Certain bits in the status register are set by the write state machine, and can only be reset by the system software. These bits can indicate various failure conditions. By allowing the system software to control the resetting of these bits, several operations may be performed (such as cumulatively programming several bytes or erasing multiple blocks in sequence). The status register may then be read to determine if an error occurred during that programming or erasure series. This adds flexibility to the way the device may be programmed or erased. To clear the status register, the Clear Status Register command is written to the CUI. Then, any other command may be issued to the CUI. Note again that before a read cycle can be initiated, a Read Array command must be written to the CUI to specify whether the read data is to come from the array, status register, or Intelligent Identifier.

4.4.4 PROGRAM MODE

Program is executed by a two-write sequence. The Program Setup command is written to the CUI followed by a second write which specifies the address and data to be programmed. The write state machine will execute a sequence of internally timed events to:

1. Program the desired bits of the addressed memory word (byte), and
2. Verify that the desired bits are sufficiently programmed.

Programming of the memory results in specific bits within a byte or word being changed to a "0".

If the user attempts to program "1"s, there will be no change of the memory cell content and no error occurs.

Similar to erasure, the status register indicates whether programming is complete. While the program sequence is executing, bit 7 of the status register is a "0". The status register can be polled by toggling either CE# or OE# to determine when the program sequence is complete. Only the Read Status Register command is valid while programming is active.

When programming is complete, the status bits, which indicate whether the program operation was successful, should be checked. If the programming operation was unsuccessful, Bit 4 of the status register is set to a "1" to indicate a Program Failure. If Bit 3 is set then V_{pp} was not within acceptable limits, and the WSM will not execute the programming sequence.

The status register should be cleared before attempting the next operation. Any CUI instruction can follow after programming is completed; however, it must be recognized that reads from the memory, status register, or Intelligent Identifier cannot be accomplished until the CUI is given the appropriate command. A Read Array command must first be given before memory contents can be read.

Figure 12 shows a system software flowchart for device byte programming operation. Figure 13 shows a similar flowchart for device word programming operation (28F400BX-only).

4.4.5 ERASE MODE

Erase of a single block is initiated by writing the Erase Setup and Erase Confirm commands to the CUI, along with the addresses, A[12:17] for the 28F400BX or A[12:18] for the 28F004BX, identifying the block to be erased. These addresses are latched internally when the Erase Confirm command is issued. Block erasure results in all bits within the block being set to "1".

The WSM will execute a sequence of internally timed events to:

1. Program all bits within the block
2. Verify that all bits within the block are sufficiently programmed
3. Erase all bits within the block and
4. Verify that all bits within the block are sufficiently erased

While the erase sequence is executing, Bit 7 of the status register is a "0".

When the status register indicates that erasure is complete, the status bits, which indicate whether the erase operation was successful, should be checked. If the erasure operation was unsuccessful, Bit 5 of the status register is set to a "1" to indicate an Erase Failure. If V_{pp} was not within acceptable limits after the Erase Confirm command is issued, the WSM will not execute an erase sequence; instead, Bits of the status register is set to a "1" to indicate an Erase Failure, and Bit 3 is set to a "1" to identify that V_{pp} supply voltage was not within acceptable limits.

The status register should be cleared before attempting the next operation. Any CUI instruction can follow after erasure is completed; however, it must be recognized that reads from the memory array, status register, or Intelligent Identifier can not be accomplished until the CUI is given the appropriate command. A Read Array command must first be given before memory contents can be read.

Figure 14 shows a system software flowchart for Block Erase operation.

4.4.5.1 Suspending and Resuming Erase

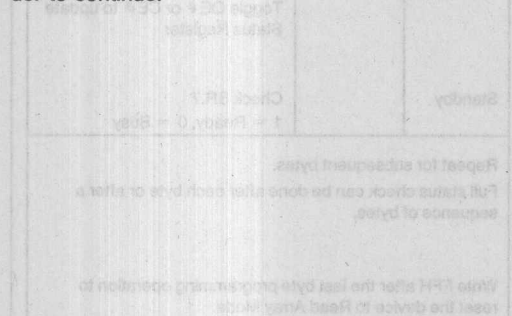
Since an erase operation typically requires 1 to 3 seconds to complete, an Erase Suspend command is provided. This allows erase-sequence interruption in order to read data from another block of the memory. Once the erase sequence is started, writing the Erase Suspend command to the CUI requests that the Write State Machine (WSM) pause the erase sequence at a predetermined point in the erase algorithm. The status register must be read to determine when the erase operation has been suspended.

At this point, a Read Array command can be written to the CUI in order to read data from blocks other than that which is being suspended. The only other valid command at this time is the Erase Resume command or Read Status Register operation.

Figure 15 shows a system software flowchart detailing the operation.

During Erase Suspend mode, the chip can go into a pseudo-standby mode by taking CE# to V_{IH} and the active current is now a maximum of 10 mA. If the chip is enabled while in this mode by taking CE# to V_{IL} , the Erase Resume command can be issued to resume the erase operation.

Upon completion of reads from any block other than the block being erased, the Erase Resume command must be issued. When the Erase Resume command is given, the WSM will continue with the erase sequence and complete erasing the block. As with the end of erase, the status register must be read, cleared, and the next instruction issued in order to continue.



Operation	Command	Comments
Standby	Check SR3 t = 100 ns	SR3 MUST be cleared, if set during a program attempt, before further attempts are allowed by the Write Status Machine.
Standby	Check SR4 t = 100 ns	SR4 is only cleared by the Clear Status Register Command, in cases where multiple bytes are programmed before full status is checked.

Figure 13. Automated Byte Programming Flowchart

4.4.6 EXTENDED CYCLING

Intel has designed extended cycling capability into its ETOX III flash memory technology. The 4-Mbit boot block flash family is designed for 100,000 program/erase cycles on each of the seven blocks. The combination of low electric fields, clean oxide processing and minimized oxide area per memory cell subjected to the tunneling electric field, results in very high cycling capability.

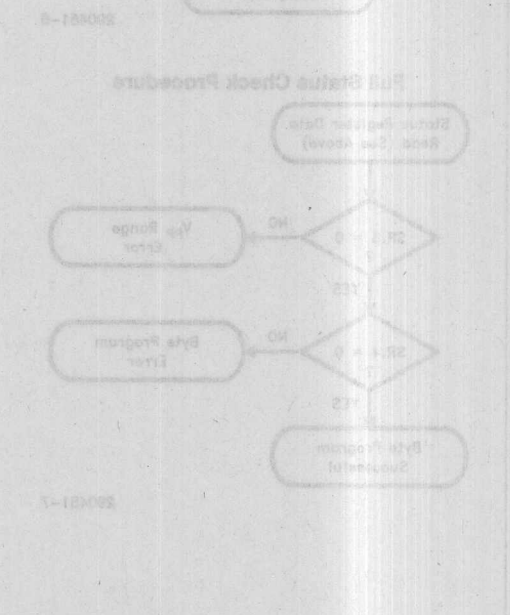
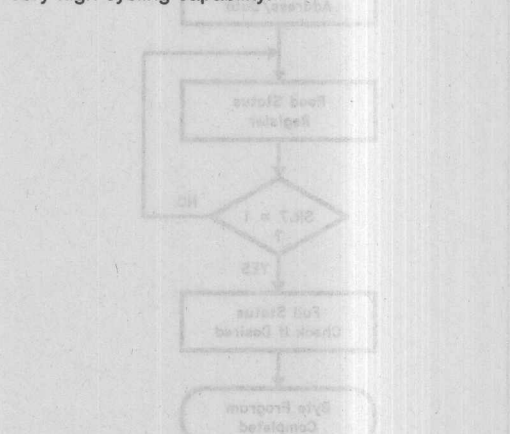


Figure 13. Automated Byte Programming Flowchart

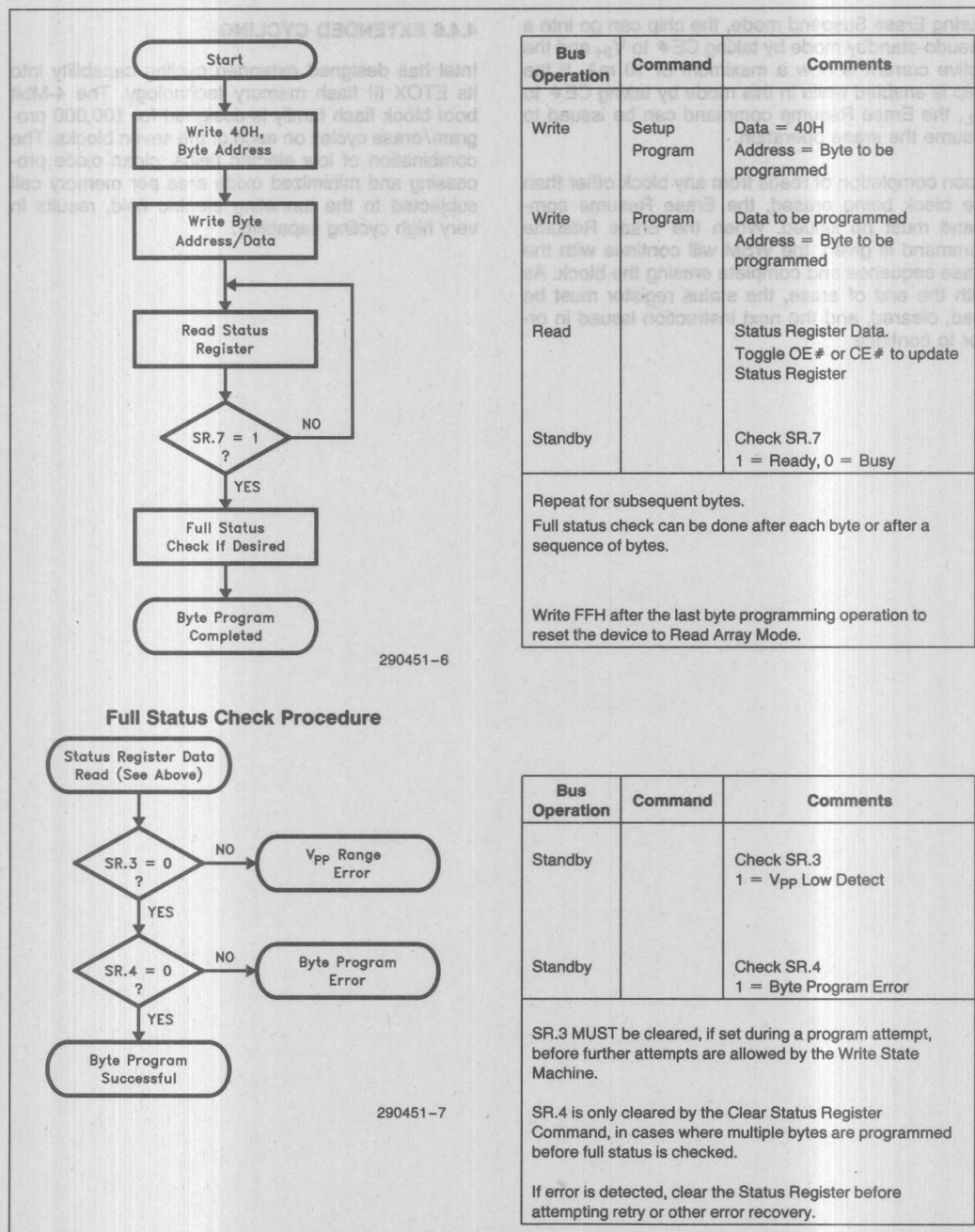
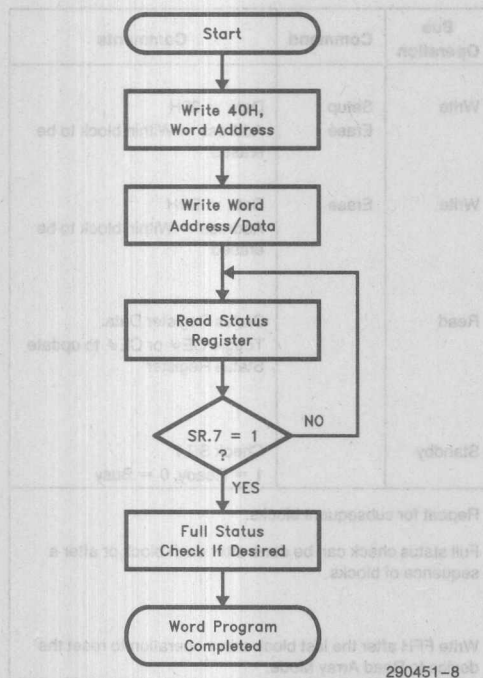
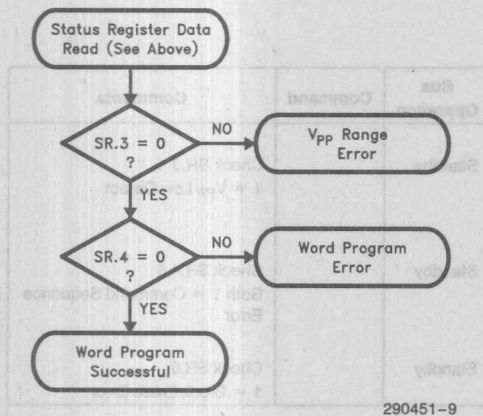


Figure 12. Automated Byte Programming Flowchart



Full Status Check Procedure



Bus Operation	Command	Comments
Write	Setup Program	Data = 40H Address = Word to be programmed
Write	Program	Data to be programmed Address = Word to be programmed
Read		Status Register Data. Toggle OE # or CE # to update Status Register
Standby		Check SR.7 1 = Ready, 0 = Busy

Repeat for subsequent words.

Full status check can be done after each word or after a sequence of words.

Write FFH after the last word programming operation to reset the device to Read Array Mode.

Bus Operation	Command	Comments
Standby		Check SR.3 1 = Vpp Low Detect
Standby		Check SR.4 1 = Word Program Error

SR.3 MUST be cleared, if set during a program attempt, before further attempts are allowed by the Write State Machine.

SR.4 is only cleared by the Clear Status Register Command, in cases where multiple words are programmed before full status is checked.

If error is detected, clear the Status Register before attempting retry or other error recovery.

Figure 13. Automated Word Programming Flowchart

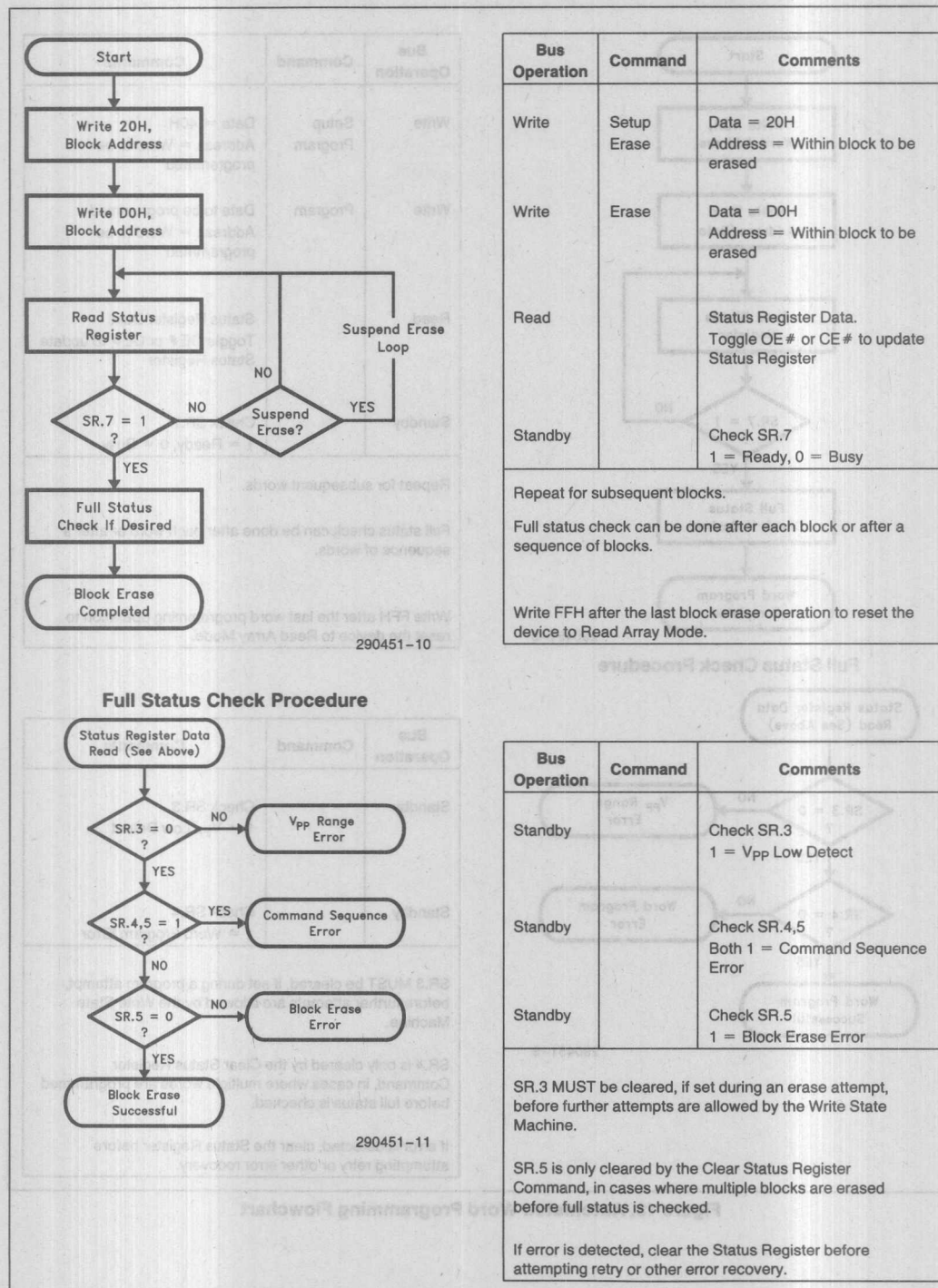


Figure 14. Automated Block Erase Flowchart

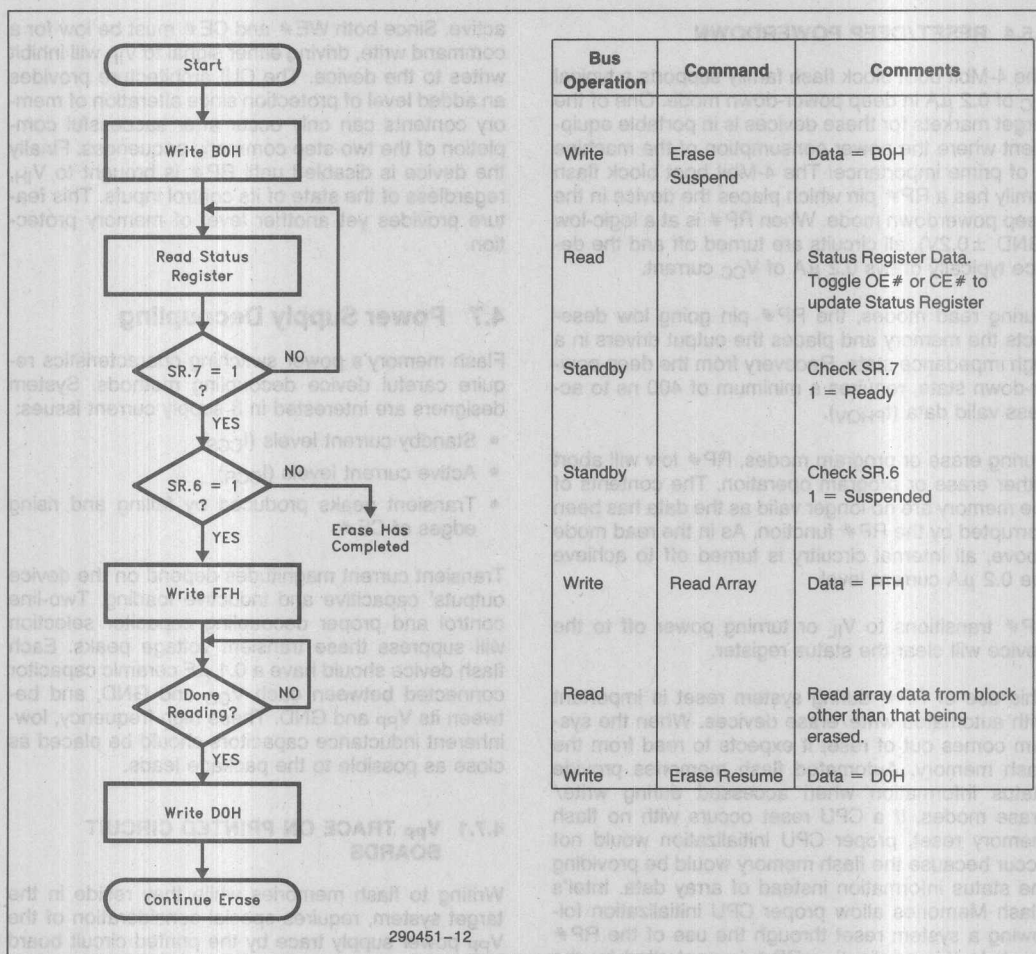


Figure 15. Erase Suspend/Resume Flowchart

4.5 Power Consumption

4.5.1 ACTIVE POWER

With CE# at a logic-low level and RP# at a logic-high level, the device is placed in the active mode. The device I_{CC} current is a maximum 60 mA at 10 MHz with TTL input signals.

4.5.2 AUTOMATIC POWER SAVINGS

Automatic Power Savings (APS) is a low power feature during active mode of operation. The 4-Mbit family of products incorporate Power Reduction Control (PRC) circuitry which basically allows the device to put itself into a low current state when it is not being accessed. After data is read from the memory array, PRC logic controls the device's power consumption by entering the APS mode where

maximum I_{CC} current is 3 mA and typical I_{CC} current is 1 mA. The device stays in this static state with outputs valid until a new location is read.

4.5.3 STANDBY POWER

With CE# at a logic-high level (V_{IH}), and the CUI in read mode, the memory is placed in standby mode where the maximum I_{CC} standby current is 100 μ A with CMOS input signals. The standby operation disables much of the device's circuitry and substantially reduces device power consumption. The outputs (DQ[0:15] or DQ[0:7]) are placed in a high-impedance state independent of the status of the OE# signal. When the 4-Mbit boot block flash family is deselected during erase or program functions, the devices will continue to perform the erase or program function and consume program or erase active power until program or erase is completed.

4.5.4 RESET/DEEP POWERDOWN

The 4-Mbit boot block flash family supports a typical I_{CC} of 0.2 μA in deep power-down mode. One of the target markets for these devices is in portable equipment where the power consumption of the machine is of prime importance. The 4-Mbit boot block flash family has a RP# pin which places the device in the deep powerdown mode. When RP# is at a logic-low (GND $\pm 0.2V$), all circuits are turned off and the device typically draws 0.2 μA of V_{CC} current.

During read modes, the RP# pin going low deselects the memory and places the output drivers in a high impedance state. Recovery from the deep power-down state, requires a minimum of 400 ns to access valid data (t_{PHQV}).

During erase or program modes, RP# low will abort either erase or program operation. The contents of the memory are no longer valid as the data has been corrupted by the RP# function. As in the read mode above, all internal circuitry is turned off to achieve the 0.2 μA current level.

RP# transitions to V_{IL} or turning power off to the device will clear the status register.

This use of RP# during system reset is important with automated write/erase devices. When the system comes out of reset it expects to read from the flash memory. Automated flash memories provide status information when accessed during write/erase modes. If a CPU reset occurs with no flash memory reset, proper CPU initialization would not occur because the flash memory would be providing the status information instead of array data. Intel's Flash Memories allow proper CPU initialization following a system reset through the use of the RP# input. In this application RP# is controlled by the same RESET# signal that resets the system CPU.

4.6 Power-up Operation

The 4-Mbit boot block flash family is designed to offer protection against accidental block erasure or programming during power transitions. Upon power-up the 4-Mbit boot block flash family is indifferent as to which power supply, V_{PP} or V_{CC} , powers-up first. Power supply sequencing is not required.

The 4-Mbit boot block flash family ensures the CUI is reset to the read mode on power-up.

In addition, on power-up the user must either drop CE# low or present a new address to ensure valid data at the outputs.

A system designer must guard against spurious writes for V_{CC} voltages above V_{LKO} when V_{PP} is

active. Since both WE# and CE# must be low for a command write, driving either signal to V_{IH} will inhibit writes to the device. The CUI architecture provides an added level of protection since alteration of memory contents can only occur after successful completion of the two-step command sequences. Finally the device is disabled until RP# is brought to V_{IH} , regardless of the state of its control inputs. This feature provides yet another level of memory protection.

4.7 Power Supply Decoupling

Flash memory's power switching characteristics require careful device decoupling methods. System designers are interested in 3 supply current issues:

- Standby current levels (I_{CCS})
- Active current levels (I_{CCR})
- Transient peaks produced by falling and rising edges of CE#.

Transient current magnitudes depend on the device outputs' capacitive and inductive loading. Two-line control and proper decoupling capacitor selection will suppress these transient voltage peaks. Each flash device should have a 0.1 μF ceramic capacitor connected between each V_{CC} and GND, and between its V_{PP} and GND. These high frequency, low-inductance capacitors should be placed as close as possible to the package leads.

4.7.1 V_{PP} TRACE ON PRINTED CIRCUIT BOARDS

Writing to flash memories while they reside in the target system, requires special consideration of the V_{PP} power supply trace by the printed circuit board designer. The V_{PP} pin supplies the flash memory cells current for programming and erasing. One should use similar trace widths and layout considerations given to the V_{CC} power supply trace. Adequate V_{PP} supply traces and decoupling will decrease spikes and overshoots.

4.7.2 V_{CC} , V_{PP} AND RP# TRANSITIONS

The CUI latches commands as issued by system software and is not altered by V_{PP} or CE# transitions or WSM actions. Its state upon power-up, after exit from deep power-down mode or after V_{CC} transitions below V_{LKO} (Lockout voltage), is Read Array mode.

After any word/byte write or block erase operation is complete and even after V_{PP} transitions down to V_{PPL} , the CUI must be reset to Read Array mode via the Read Array command when accesses to the flash memory are desired.

ABSOLUTE MAXIMUM RATINGS*

Commercial Operating Temperature	
During Read	0°C to 70°C(1)
During Block Erase	
and Word/Byte Write	0°C to 70°C
Temperature Under Bias	-10°C to +80°C
Extended Operating Temperature	
During Read	-40°C to +85°C
During Block Erase	
and Word/Byte Write	-40°C to +85°C
Temperature Under Bias	-40°C to +85°C
Storage Temperature	
	-65°C to +125°C
Voltage on Any Pin	
(except V _{CC} , V _{PP} , A _g and RP#)	
with Respect to GND	-2.0V to +7.0V(2)
Voltage on Pin RP# or Pin A _g	
with Respect to GND	-2.0V to +13.5V(2,3)
V _{PP} Program Voltage with Respect	
to GND during Block Erase	
and Word/Byte Write	-2.0V to +14.0V(2,3)
V _{CC} Supply Voltage	
with Respect to GND	-2.0V to +7.0V(2)
Output Short Circuit Current	
	100 mA(4)

NOTES:

- Operating temperature is for commercial product defined by this specification.
- Minimum DC voltage is -0.5V on input/output pins. During transitions, this level may undershoot to -2.0V for periods <20 ns.
- Maximum DC voltage on input/output pins is V_{CC} + 0.5V which, during transitions, may overshoot to V_{CC} + 2.0V for periods <20 ns.
- Maximum DC voltage on V_{PP} may overshoot to +14.0V for periods <20 ns. Maximum DC voltage on RP# or A_g may overshoot to 13.5V for periods < 20 ns.
- Output shorted for no more than one second. No more than one output shorted at a time.
- 10% V_{CC} specifications reference the 28F400BX-60/28F004BX-60 in their standard test configuration, and the 28F400BX-80/28F004BX-80.
- 5% V_{CC} specifications reference the 28F400BX-60/28F004BX-60 in their high speed test configuration.

OPERATING CONDITIONS

Symbol	Parameter	Notes	Min	Max	Units
T _A	Operating Temperature		0	70	°C
V _{CC}	V _{CC} Supply Voltage (10%)	5	4.50	5.50	V
V _{CC}	V _{CC} Supply Voltage (5%)	6	4.75	5.25	V

DC CHARACTERISTICS

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Condition
I _{LI}	Input Load Current	1			±1.0	μA	V _{CC} = V _{CC} Max V _{IN} = V _{CC} or GND
I _{LO}	Output Leakage Current	1			±10	μA	V _{CC} = V _{CC} Max V _{OUT} = V _{CC} or GND

NOTICE: This data sheet contains information on products in the sampling and initial production phases of development. The specifications are subject to change without notice. Verify with your local Intel Sales office that you have the latest data sheet before finalizing a design.

***WARNING:** Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.

DC CHARACTERISTICS (Continued)

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Conditions
I _{CCS}	V _{CC} Standby Current	1, 3			1.5	mA	V _{CC} = V _{CC} Max CE# = RP# = V _{IH}
					100	μA	V _{CC} = V _{CC} Max CE# = RP# = V _{CC} ± 0.2V 28F400BX: BYTE# = V _{CC} ± 0.2V or GND
I _{CCD}	V _{CC} Deep Powerdown Current	1		0.20	1.2	μA	RP# = GND ± 0.2V
I _{CCR}	V _{CC} Read Current for 28F400BX Word-Wide and Byte-Wide Mode and 28F004BX Byte-Wide Mode	1, 5, 6		20	55	mA	V _{CC} = V _{CC} Max, CE# = GND f = 10 MHz, I _{OUT} = 0 mA CMOS Inputs
				20	60	mA	V _{CC} = V _{CC} Max, CE# = V _{IL} f = 10 MHz, I _{OUT} = 0 mA TTL Inputs
I _{CCW}	V _{CC} Word/Byte Write Current	1, 4			65	mA	Word or Byte Write in Progress
I _{CCE}	V _{CC} Block Erase Current	1, 4			30	mA	Block Erase in Progress
I _{CCES}	V _{CC} Erase Suspend Current	1, 2		5	10	mA	Block Erase Suspended, CE# = V _{IH}
I _{PPS}	V _{PP} Standby Current	1			± 15	μA	V _{PP} ≤ V _{CC}
I _{PPD}	V _{PP} Deep PowerDown Current	1			5.0	μA	RP# = GND ± 0.2V
I _{PPR}	V _{PP} Read Current	1			200	μA	V _{PP} > V _{CC}
I _{PPW}	V _{PP} Word Write Current	1			40	mA	V _{PP} = V _{PPH} Word Write in Progress
I _{PPW}	V _{PP} Byte Write Current	1			30	mA	V _{PP} = V _{PPH} Byte Write in Progress
I _{PPE}	V _{PP} Block Erase Current	1			30	mA	V _{PP} = V _{PPH} Block Erase in Progress
I _{PPES}	V _{PP} Erase Suspend Current	1			200	μA	V _{PP} = V _{PPH} Block Erase Suspended
I _{RP#}	RP# Boot Block Unlock Current	1, 4			500	μA	RP# = V _{HH}
I _{ID}	A ₉ Intelligent Identifier Current	1, 4			500	μA	A ₉ = V _{ID}
V _{ID}	A ₉ Intelligent Identifier Voltage		11.5		13.0	V	
V _{IL}	Input Low Voltage		-0.5		0.8	V	
V _{IH}	Input High Voltage		2.0		V _{CC} + 0.5	V	
V _{OL}	Output Low Voltage				0.45	V	V _{CC} = V _{CC} Min I _{OL} = 5.8 mA

DC CHARACTERISTICS (Continued)

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Conditions
V _{OH}	Output High Voltage		2.4			V	V _{CC} = V _{CC} Min I _{OH} = -2.5 mA
V _{PPL}	V _{PP} during Normal Operations	3	0.0		6.5	V	
V _{PPH}	V _{PP} during Erase/Write Operations	7	11.4	12.0	12.6	V	
V _{PPH}	V _{PP} during Erase/Write Operations	8	10.8	12.0	13.2	V	
V _{LKO}	V _{CC} Erase/Write Lock Voltage		2.0			V	
V _{HH}	RP# Unlock Voltage		11.5		13.0	V	Boot Block Write/Erase

EXTENDED TEMPERATURE OPERATING CONDITIONS

Symbol	Parameter	Notes	Min	Max	Unit
T _A	Operating Temperature		-40	85	°C
V _{CC}	V _{CC} Supply Voltage (10%)	5	4.50	5.50	V

DC CHARACTERISTICS: EXTENDED TEMPERATURE OPERATION

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Conditions
I _{LI}	Input Load Current	1			±1.0	μA	V _{CC} = V _{CC} Max V _{IN} = V _{CC} or GND
I _{LO}	Output Leakage Current	1			±10	μA	V _{CC} = V _{CC} Max V _{OUT} = V _{CC} or GND
I _{CCS}	V _{CC} Standby Current	1, 3			1.5	mA	V _{CC} = V _{CC} Max CE# = RP# = V _{IH} RP#
					100	μA	V _{CC} = V _{CC} Max CE# = RP# = V _{CC} ±0.2V 28F400BX: BYTE# = V _{CC} ±0.2V or GND
I _{CCD}	V _{CC} Deep Power-Down Current	1		0.20	8	μA	RP# = GND ±0.2V

DC CHARACTERISTICS: EXTENDED TEMPERATURE OPERATION (Continued)

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Conditions
I_{CCR}	V_{CC} Read Current for 28F400BX Word-Wide and Byte-Wide Mode 28F004BX Byte-Wide Mode	1, 5, 6			60	mA	$V_{CC} = V_{CC} \text{ Max}$, $CE\# = GND$ $f = 10 \text{ MHz}$, $I_{OUT} = 0 \text{ mA}$ CMOS Inputs
					65	mA	$V_{CC} = V_{CC} \text{ Max}$, $CE\# = V_{IL}$ $f = 10 \text{ MHz}$, $I_{OUT} = 0 \text{ mA}$ TTL Inputs
I_{CCW}	V_{CC} Word Write Current	1			70	mA	Word Write in Progress
I_{CCE}	V_{CC} Block Erase Current	1			40	mA	Block Erase in Progress
I_{CCES}	V_{CC} Erase Suspend Current	1, 2		5	10	mA	Block Erase Suspended, $CE\# = V_{IH}$
I_{PPS}	V_{PP} Standby Current	1			± 15	μA	$V_{PP} \leq V_{CC}$
I_{PPD}	V_{CC} Deep Power-Down Current	1			5.0	μA	$RP\# = GND \pm 0.2V$
I_{PPR}	V_{PP} Read Current	1			200	μA	$V_{PP} > V_{CC}$
I_{PPW}	V_{PP} Word Write Current	1			40	mA	$V_{PP} = V_{PPH}$ Word Write in Progress
I_{PPW}	V_{PP} Byte Write Current	1			30	mA	$V_{PP} = V_{PPH}$ Byte Write in Progress

I_{CC}	V_{CC} Read Current	1			± 10	μA	$V_{CC} = V_{CC} \text{ Max}$ $V_{IN} = V_{CC} \text{ or } GND$
I_{CC}	V_{CC} Read Current	1			± 10	μA	$V_{CC} = V_{CC} \text{ Max}$ $V_{OUT} = V_{CC} \text{ or } GND$
I_{CC}	V_{CC} Standby Current	1			± 10	μA	$V_{CC} = V_{CC} \text{ Max}$ $CE\# = RP\# = V_{CC} \text{ or } GND$
I_{CC}	V_{CC} Standby Current	1			± 10	μA	$V_{CC} = V_{CC} \text{ Max}$ $CE\# = RP\# = V_{CC} \pm 0.2V$ 28F004BX BYTE# = $V_{CC} \pm 0.2V$ or GND
I_{CC}	V_{CC} Deep Power-Down Current	1			± 0.2	μA	$RP\# = GND \pm 0.2V$

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Conditions
I _{PPE}	V _{PP} Block Erase Current	1			30	mA	V _{PP} = V _{PPH} Block Erase in Progress
I _{PPES}	V _{PP} Erase Suspend Current	1			200	μA	V _{PP} = V _{PPH} Block Erase Suspended
I _{RP#}	RP# Boot Block Unlock Current	1, 4			500	μA	RP# = V _{HH}
I _{ID}	A ₉ Intelligent Identifier Current	1			500	μA	A ₉ = V _{ID}
V _{ID}	A ₉ Intelligent Identifier Current		11.5		13.0	V	
V _{IL}	Input Low Voltage		-0.5		0.8	V	
V _{IH}	Input High Voltage		2.0		V _{CC} + 0.5	V	
V _{OL}	Output Low Voltage				0.45	V	V _{CC} = V _{CC} Min I _{OL} = 5.8 mA
V _{OH}	Output High Voltage		2.4			V	V _{CC} = V _{CC} Min I _{OH} = -2.5 mA
V _{PPL}	V _{PP} during Normal Operations	3	0.0		6.5	V	
V _{PPH}	V _{PP} during Erase/Write Operations	7	11.4	12.0	12.6	V	
V _{PPH}	V _{PP} during Erase/Write Operations	8	10.8	12.0	13.2	V	
V _{LKO}	V _{CC} Erase/Write Lock Voltage		2.0			V	
V _{HH}	RP# Unlock Voltage		11.5		13.0	V	Boot Block Write/Erase

4

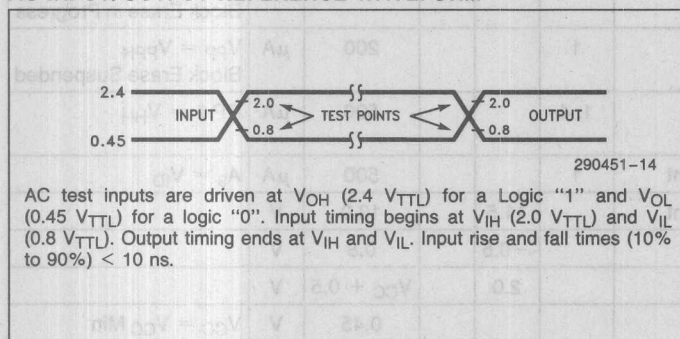
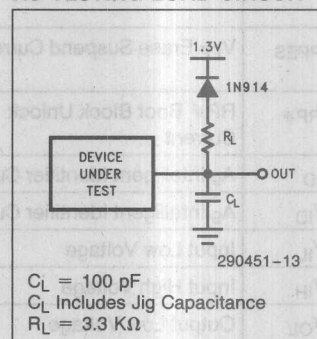
CAPACITANCE(4) T_A = 25°C, f = 1 MHz

Symbol	Parameter	Typ	Max	Unit	Conditions
C _{IN}	Input Capacitance	6	8	pF	V _{IN} = 0V
C _{OUT}	Output Capacitance	10	12	pF	V _{OUT} = 0V

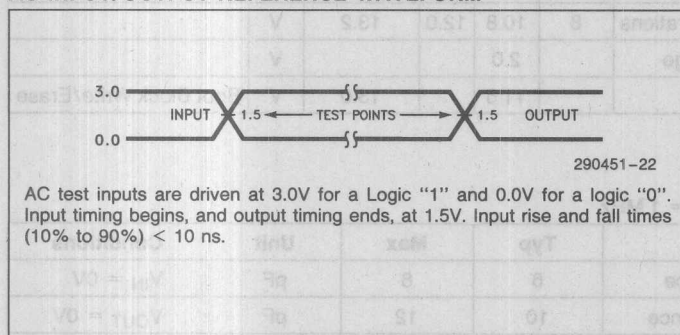
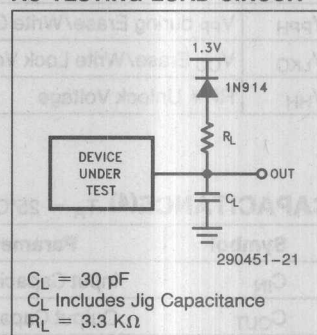
NOTES:

1. All currents are in RMS unless otherwise noted. Typical values at V_{CC} = 5.0V, V_{PP} = 12.0V, T = 25°C. These currents are valid for all product versions (packages and speeds).
2. I_{CCES} is specified with the device deselected. If the device is read while in Erase Suspend Mode, current draw is the sum of I_{CCES} and I_{CCR}.
3. Block Erases and Word/Byte Writes are inhibited when V_{PP} = V_{PPL} and not guaranteed in the range between V_{PPH} and V_{PPL}.
4. Sampled, not 100% tested.
5. Automatic Power Savings (APS) reduces I_{CCR} to less than 1 mA typical in static operation.
6. CMOS Inputs are either V_{CC} ± 0.2V or GND ± 0.2V. TTL Inputs are either V_{IL} or V_{IH}.
7. V_{PP} = 12.0V ± 5% for applications requiring 100,000 block erase cycles.
8. V_{PP} = 12.0V ± 10% for applications requiring wider V_{PP} tolerances at 100 block erase cycles.
9. For the 28F004BX address pin A₁₀ follows the C_{OUT} capacitance numbers.
10. I_{CCR} typical is 25 mA for X16 Active Read Current.

STANDARD TEST CONFIGURATION(1)

STANDARD
AC INPUT/OUTPUT REFERENCE WAVEFORMSTANDARD
AC TESTING LOAD CIRCUIT

HIGH SPEED TEST CONFIGURATION(2)

HIGH SPEED
AC INPUT/OUTPUT REFERENCE WAVEFORMHIGH SPEED
AC TESTING LOAD CIRCUIT

NOTES:

1. Testing characteristics for 28F400BX-60/28F004BX-60 in standard test configuration and 28F400BX-80/28F004BX-80.
2. Testing characteristics for 28F400BX-60/28F004BX-60 in high speed test configuration.

AC CHARACTERISTICS—Read Only Operations(1)

Versions		V _{CC} ± 5%		28F400BX-60(4) 28F004BX-60(4)						Unit
		V _{CC} ± 10%				28F400BX-60(5) 28F004BX-60(5)		28F400BX-80(5) 28F004BX-80(5)		
Symbol		Parameter	Notes	Min	Max	Min	Max	Min	Max	
t _{AVAV}	t _{RC}	Read Cycle Time		60		70		80		ns
t _{AVQV}	t _{ACC}	Address to Output Delay			60		70		80	ns
t _{ELQV}	t _{CE}	CE # to Output Delay	2		60		70		80	ns
t _{PHQV}	t _{PWH}	RP # High to Output Delay			300		300		300	ns
t _{GLQV}	t _{OE}	OE # to Output Delay	2		30		35		40	ns
t _{ELQX}	t _{LZ}	CE # to Output Low Z	3	0		0		0		ns
t _{EHQZ}	t _{HZ}	CE # High to Output High Z	3		20		25		30	ns
t _{GLQX}	t _{OLZ}	OE # to Output Low Z	3	0		0		0		ns
t _{GHQZ}	t _{DF}	OE # High to Output High Z	3		20		25		30	ns
	t _{OH}	Output Hold from Addresses, CE # or OE # Change, Whichever is First	3	0		0		0		ns
t _{ELFL}	t _{ELFH}	CE # to BYTE # Switching Low or High	3		5		5		5	ns
t _{FHQV}		BYTE # Switching High to Valid Output Delay	3, 6		60		70		80	ns
t _{FLQZ}		BYTE # Switching Low to Output High Z	3		20		25		30	ns

NOTES:

1. See AC Input/Output Reference Waveform for timing measurements.
2. OE # may be delayed up to $t_{CE}-t_{OE}$ after the falling edge of CE # without impact on t_{CE} .
3. Sampled, not 100% tested.
4. See High Speed Test Configuration.
5. See Standard Test Configuration.
6. t_{FLQV} , BYTE # switching low to valid output delay, will be equal to t_{AVQV} , measured from the time DQ₁₅/A₁ becomes valid.

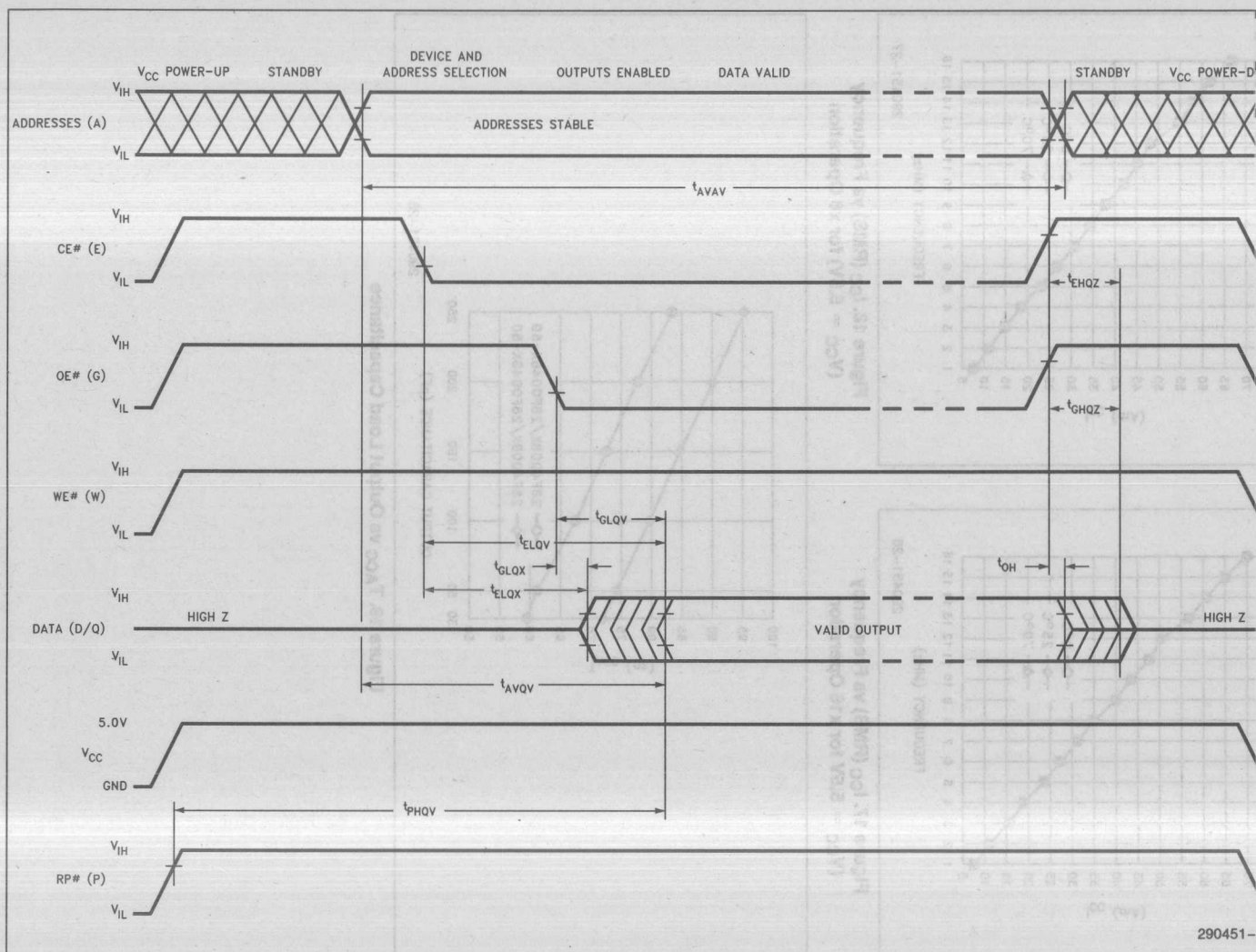
EXTENDED TEMPERATURE OPERATION AC CHARACTERISTICS—Read Only Operations⁽¹⁾

Versions			T28F400BX-80 ⁽⁴⁾ T28F004BX-80		Unit
Symbol	Parameter	Notes	Min	Max	
t _{AVAV}	t _{RC}	Read Cycle Time	80		ns
t _{AVQV}	t _{ACC}	Address to Output Delay		80	ns
t _{ELQV}	t _{CE}	CE # to Output Delay		80	ns
t _{PHQV}	t _{PWH}	RP # High to Output Delay		300	ns
t _{GLQV}	t _{OE}	OE # to Output Delay	2	40	ns
t _{ELQX}	t _{LZ}	CE # to Output Low Z	0		ns
t _{EHQZ}	t _{HZ}	CE # High to Output High Z		30	ns
t _{GLQX}	t _{OLZ}	OE # to Output Low Z	3	0	ns
t _{GHQZ}	t _{DF}	OE # High to Output High Z	3	30	ns
	t _{OH}	Output Hold from Addresses, CE # or OE # Change, Whichever is First	3	0	ns
t _{ELFL} t _{ELFH}		CE # to BYTE # Switching Low or High	3	5	ns
t _{FHQV}		BYTE # Switching High to Valid Output Delay	3, 5	80	ns
t _{FLQZ}		BYTE # Switching Low to Output High Z	3	30	ns

NOTES:

1. See AC Input/Output Reference Waveform for timing measurements.
2. OE # may be delayed up to t_{CE}-t_{OE} after the falling edge of CE # without impact on t_{CE}.
3. Sampled, not 100% tested.
4. See Standard Test Configuration.
5. t_{FLQV}, BYTE # switching low to valid output delay, will be equal to t_{AVQV} from the time DQ₁₅/A₁ becomes valid.

Figure 16. A.C. Waveforms for Read Operations



290451-

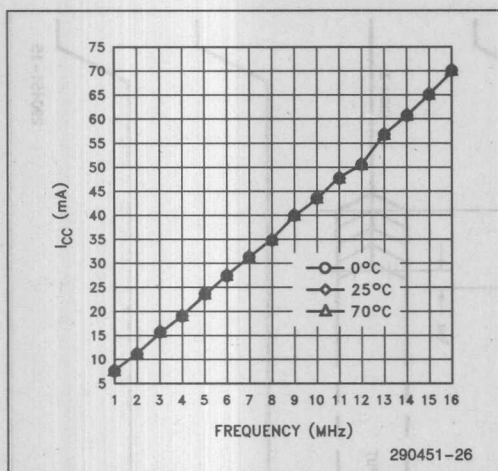


Figure 17. I_{CC} (RMS) vs Frequency
(V_{CC} = 5.5V for x16 Operation)

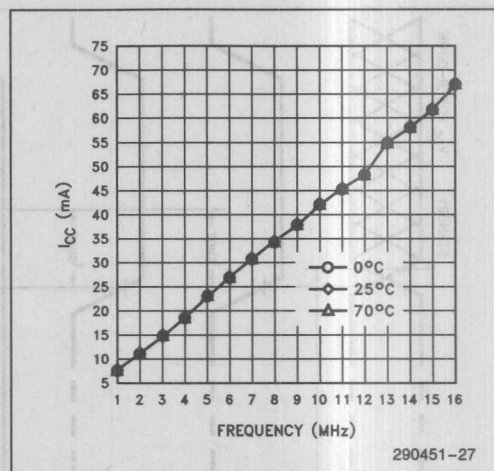


Figure 18. I_{CC} (RMS) vs Frequency
(V_{CC} = 5.5V) for x8 Operation

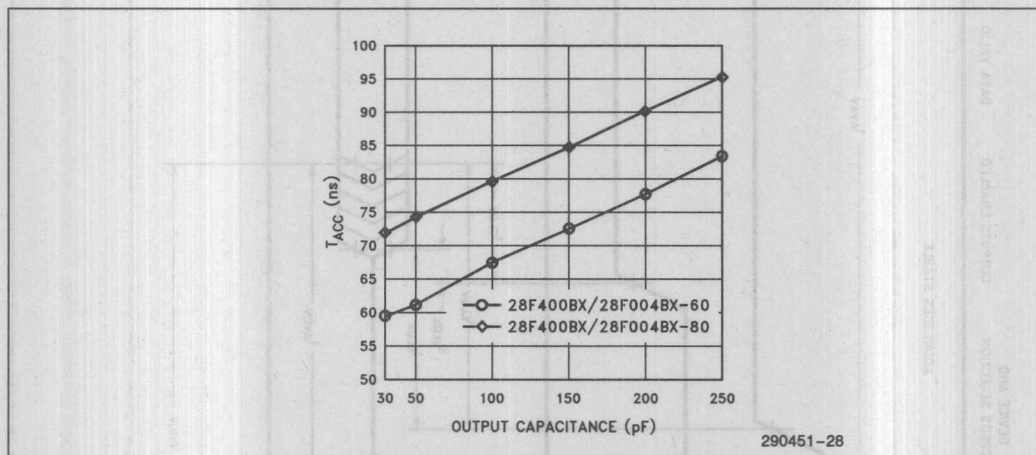


Figure 19. T_{ACC} vs Output Load Capacitance

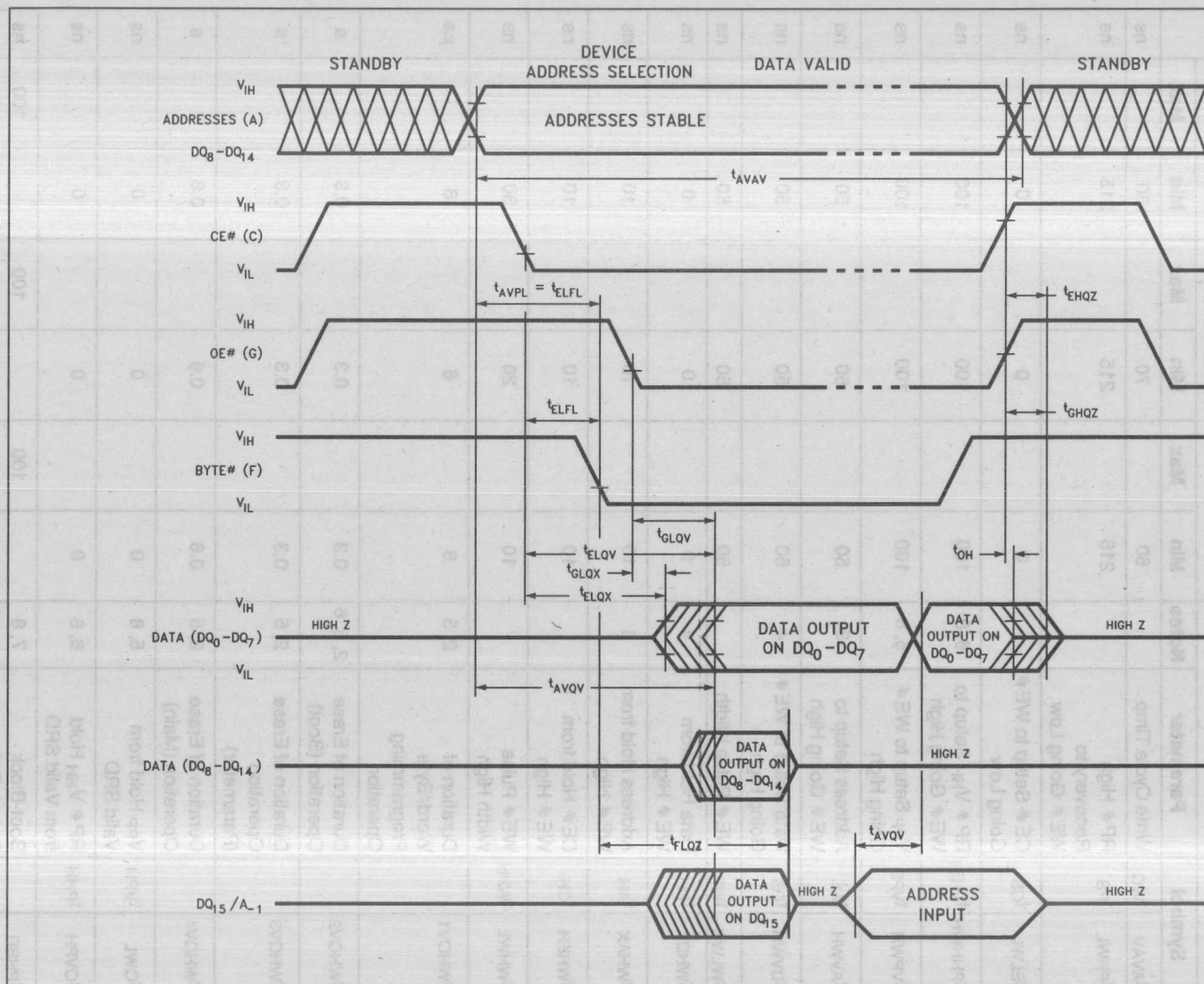


Figure 20. BYTE # Timing Diagram for Both Read and Write Operations for 28F400BX

AC CHARACTERISTICS—WE # Controlled Write Operations⁽¹⁾

Versions		VCC ± 5%		28F400BX-60 ⁽⁹⁾ 28F004BX-60 ⁽⁹⁾						Unit
		VCC ± 10%				28F400BX-60 ⁽¹⁰⁾ 28F004BX-60 ⁽¹⁰⁾		28F400BX-80 ⁽¹⁰⁾ 28F004BX-80 ⁽¹⁰⁾		
Symbol		Parameter	Notes	Min	Max	Min	Max	Min	Max	
tAVAV	tWC	Write Cycle Time		60		70		80		ns
tPHWL	tPS	RP # High Recovery to WE # Going Low		215		215		215		ns
tELWL	tCS	CE # Setup to WE # Going Low		0		0		0		ns
tPHHWH	tPHS	RP # V _{HH} Setup to WE # Going High	6, 8	100		100		100		ns
tVPWH	tVPS	V _{PP} Setup to WE # Going High	5, 8	100		100		100		ns
tAVWH	tAS	Address Setup to WE # Going High	3	50		50		50		ns
tDVWH	tDS	Data Setup to WE # Going High	4	50		50		50		ns
tWLWH	tWP	WE # Pulse Width		50		50		50		ns
tWHDX	tDH	Data Hold from WE # High	4	0		0		0		ns
tWHAX	tAH	Address Hold from WE # High	3	10		10		10		ns
tWHEH	tCH	CE # Hold from WE # High		10		10		10		ns
tWHWL	tWPH	WE # Pulse Width High		10		20		30		ns
tWHQV1		Duration of Word/Byte Programming Operation	2, 5	6		6		6		μs
tWHQV2		Duration of Erase Operation (Boot)	2, 5, 6	0.3		0.3		0.3		s
tWHQV3		Duration of Erase Operation (Parameter)	2, 5	0.3		0.3		0.3		s
tWHQV4		Duration of Erase Operation (Main)	2, 5	0.6		0.6		0.6		s
tQWL	tVPH	V _{PP} Hold from Valid SRD	5, 8	0		0		0		ns
tQVPH	tPHH	RP # V _{HH} Hold from Valid SRD	6, 8	0		0		0		ns
tPHBR		Boot-Block Relock Delay	7, 8		100		100		100	ns

AC CHARACTERISTICS—WE # Controlled Write Operations⁽¹⁾ (Continued)

NOTES:

1. Read timing characteristics during write and erase operations are the same as during read-only operations. Refer to AC characteristics during Read Mode.
2. The on-chip WSM completely automates program/erase operations; program/erase algorithms are now controlled internally which includes verify and margining operations.
3. Refer to command definition table for valid A_{IN} .
4. Refer to command definition table for valid D_{IN} .
5. Program/Erase durations are measured to valid SRD data (successful operation, $SR.7 = 1$).
6. For Boot Block Program/Erase, $RP\#$ should be held at V_{HH} until operation completes successfully.
7. Time t_{PBBR} is required for successful relocking of the Boot Block.
8. Sampled but not 100% tested.
9. See High Speed Test Configuration.
10. See Standard Test Configuration.

BLOCK ERASE AND WORD/BYTE WRITE PERFORMANCE $V_{PP} = 12.0V \pm 5\%$

Parameter	Notes	28F400BX-60 28F004BX-60			28F400BX-80 28F004BX-80			Unit
		Min	Typ ⁽¹⁾	Max	Min	Typ ⁽¹⁾	Max	
Boot/Parameter Block Erase Time	2		1.0	7		1.0	7	s
Main Block Erase Time	2		2.4	14		2.4	14	s
Main Block Byte Program Time	2		1.2	4.2		1.2	4.2	s
Main Block Word Program Time	2		0.6	2.1		0.6	2.1	s

NOTES:

1. 25°C
2. Excludes System-Level Overhead.

BLOCK ERASE AND WORD/BYTE WRITE PERFORMANCE $V_{PP} = 12.0V \pm 10\%$

Parameter	Notes	28F400BX-60 28F004BX-60			28F400BX-80 28F004BX-80			Unit
		Min	Typ ⁽¹⁾	Max	Min	Typ ⁽¹⁾	Max	
Boot/Parameter Block Erase Time	2		5.8	40		5.8	40	s
Main Block Erase Time	2		14	60		14	60	s
Main Block Byte Program Time	2		6.0	20		6.0	20	s
Main Block Word Program Time	2		3.0	10		3.0	10	s

NOTES:

1. 25°C
2. Excludes System-Level Overhead.

EXTENDED TEMPERATURE OPERATION **AC CHARACTERISTICS—WE # Controlled Write Operations⁽¹⁾**

Versions ⁽⁴⁾			T28F400BX-80 ⁽⁹⁾ T28F004BX-80 ⁽⁹⁾		Unit
Symbol	Parameter	Notes	Min	Max	
t _{AVAV}	t _{WC}	Write Cycle Time	80		ns
t _{PHWL}	t _{PS}	RP # High Recovery to WE # Going Low	220		ns
t _{ELWL}	t _{CS}	CE # Setup to WE # Going Low	0		ns
t _{PHWH}	t _{PHS}	RP # V _{HH} Setup to WE # Going High	100		ns
t _{VPWH}	t _{VPS}	V _{PP} Setup to WE # Going High	100		ns
t _{AVWH}	t _{AS}	Address Setup to WE # Going High	60		ns
t _{DVWH}	t _{DS}	Data Setup to WE # Going High	60		ns
t _{WLWH}	t _{WP}	WE # Pulse Width	60		ns
t _{WHDx}	t _{DH}	Data Hold from WE # High	0		ns
t _{WHAX}	t _{AH}	Address Hold from WE # High	10		ns
t _{WHEH}	t _{CH}	CE # Hold from WE # High	10		ns
t _{WHWL}	t _{WPH}	WE # Pulse Width High	20		ns
t _{WHQV1}		Duration of Word/Byte Programming Operation	7		μs
t _{WHQV2}		Duration of Erase Operation (Boot)	0.4		s
t _{WHQV3}		Duration of Erase Operation (Parameter)	0.4		s
t _{WHQV4}		Duration of Erase Operation (Main)	0.7		s
t _{QWL}	t _{VPH}	V _{PP} Hold from Valid SRD	0		ns
t _{QVPH}	t _{PHH}	RP # V _{HH} Hold from Valid SRD	0		ns
t _{PHBR}		Boot-Block Relock Delay	7, 8	100	ns

8	40	2.8	40	2.8	2	Boot/Parameter
8	80	1.4	80	1.4	2	Block Erase Time
8	20	0.0	20	0.0	2	Main Block Erase Time
8	10	3.0	10	3.0	2	Main Block Word Programming Time

NOTES:
1. 25°C
2. Excludes System-Level Overhead

NOTES:

1. Read timing characteristics during write and erase operations are the same as during read-only operations. Refer to AC characteristics during Read Mode.
2. The on-chip WSM completely automates program/erase operations; program/erase algorithms are now controlled internally which includes verify and margining operations.
3. Refer to command definition table for valid A_{IN} .
4. Refer to command definition table for valid D_{IN} .
5. Program/Erase durations are measured to valid SRD data (successful operation, SR.7 = 1).
6. For Boot Block Program/Erase, RP# should be held at V_{HH} until operation completes successfully.
7. Time t_{PHBR} is required for successful reloading of the Boot Block.
8. Sampled but not 100% tested.
9. See Standard Test Configuration.

EXTENDED TEMPERATURE OPERATION**BLOCK ERASE AND WORD/BYTE WRITE PERFORMANCE** $V_{PP} = 12.0V \pm 5\%$

Parameter	Notes	T28F400BX-80 T28F004BX-80			Unit
		Min	Typ(1)	Max	
Boot/Parameter Block Erase Time	2		1.5	10.5	s
Main Block Erase Time	2		3.0	18	s
Main Block Byte Program Time	2		1.4	5.0	s
Main Block Word Program Time	2		0.7	2.5	s

NOTES:

1. 25°C
2. Excludes System-Level Overhead.



4-44

AC CHARACTERISTICS—CE #-CONTROLLED WRITE OPERATIONS(1, 9)

Versions		V _{CC} ± 5%		28F400BX-60(10) 28F004BX-60(10)						Unit
		V _{CC} ± 10%				28F400BX-60(11) 28F004BX-60(11)		28F400BX-80(11) 28F004BX-80(11)		
Symbol		Parameter	Notes	Min	Max	Min	Max	Min	Max	
t _{AVAV}	t _{WC}	Write Cycle Time		60		70		80		ns
t _{PHL}	t _{PS}	RP# High Recovery to CE# Going Low		215		215		215		ns
t _{WLEL}	t _{WS}	WE# Setup to CE# Going Low		0		0		0		ns
t _{PHHEH}	t _{PHS}	RP# V _{HH} Setup to CE# Going High	6, 8	100		100		100		ns
t _{VPEH}	t _{VPS}	V _{PP} Setup to CE# Going High	5, 8	100		100		100		ns
t _{AVEH}	t _{AS}	Address Setup to CE# Going High	3	50		50		50		ns
t _{DVEH}	t _{DS}	Data Setup to CE# Going High	4	50		50		50		ns
t _{LEH}	t _{CP}	CE# Pulse Width		50		50		50		ns
t _{EHDX}	t _{DH}	Data Hold from CE# High	4	0		0		0		ns
t _{EHAX}	t _{AH}	Address Hold from CE# High	3	10		10		10		ns
t _{EHWH}	t _{WH}	WE# Hold from CE# High		10		10		10		ns
t _{EH}	t _{CPH}	CE# Pulse Width High		10		20		30		ns
t _{EHQV1}		Duration of Word/Byte Programming Operation	2, 5	6		6		6		μs
t _{EHQV2}		Duration of Erase Operation (Boot)	2, 5, 6	0.3		0.3		0.3		s
t _{EHQV3}		Duration of Erase Operation (Parameter)	2, 5	0.3		0.3		0.3		s
t _{EHQV4}		Duration of Erase Operation (Main)	2, 5	0.6		0.6		0.6		s
t _{QWL}	t _{VPH}	V _{PP} Hold from Valid SRD	5, 8	0		0		0		ns
t _{QVPH}	t _{PHH}	RP# V _{HH} Hold from Valid SRD	6, 8	0		0		0		ns
t _{PHBR}		Boot-Block Relock Delay	7		100		100		100	ns

AC CHARACTERISTICS—CE #-CONTROLLED WRITE OPERATIONS(1, 9) (Continued)**NOTES:**

1. Chip-Enable Controlled Writes: Write operations are driven by the valid combination of CE# and WE# in systems where CE# defines the write pulse-width (within a longer WE# timing waveform), all set-up, hold and inactive WE# times should be measured relative to the CE# waveform.

2, 3, 4, 5, 6, 7, 8: Refer to AC Characteristics notes for WE#-Controlled Write Operations.

9. Read timing characteristics during write and erase operations are the same as during read-only operations. Refer to AC Characteristics during Read Mode.

10. See High Speed Test Configuration.

11. See Standard Test Configuration.

EXTENDED TEMPERATURE OPERATION**AC CHARACTERISTICS—CE #-CONTROLLED WRITE OPERATIONS(1, 9)**

Versions			T28F400BX-80(10) T28F004BX-80(10)		Unit
Symbol	Parameter	Notes	Min	Max	
t _{AVAV}	t _{WC}	Write Cycle Time	80		ns
t _{PHL}	t _{PS}	RP# High Recovery to CE# Going Low	220		ns
t _{WLEL}	t _{WS}	WE# Setup to CE# Going Low	0		ns
t _{PHHEH}	t _{PHS}	RP# V _{HH} Setup to CE# Going High	100		ns
t _{VPEH}	t _{VPS}	V _{PP} Setup to CE# Going High	100		ns
t _{AVEH}	t _{AS}	Address Setup to CE# Going High	60		ns
t _{DVEH}	t _{DS}	Data Setup to CE# Going High	60		ns
t _{ELEH}	t _{CP}	CE# Pulse Width	60		ns
t _{EHDX}	t _{DH}	Data Hold from CE# High	0		ns
t _{EHAX}	t _{AH}	Address Hold from CE# High	10		ns
t _{EHWH}	t _{WH}	WE# Hold from CE# High	10		ns
t _{EHEL}	t _{CPH}	CE# Pulse Width High	20		ns
t _{EHQV1}		Duration of Word/Byte Programming Operation	7		μs
t _{EHQV2}		Duration of Erase Operation (Boot)	0.4		s
t _{EHQV3}		Duration of Erase Operation (Parameter)	0.4		s
t _{EHQV4}		Duration of Erase Operation (Main)	0.7		s
t _{QWL}	t _{VPH}	V _{PP} Hold from Valid SRD	0		ns
t _{QVPH}	t _{PHH}	RP# V _{HH} Hold from Valid SRD	0		ns
t _{PHBR}		Boot-Block Relock Delay	7	100	ns

NOTES:

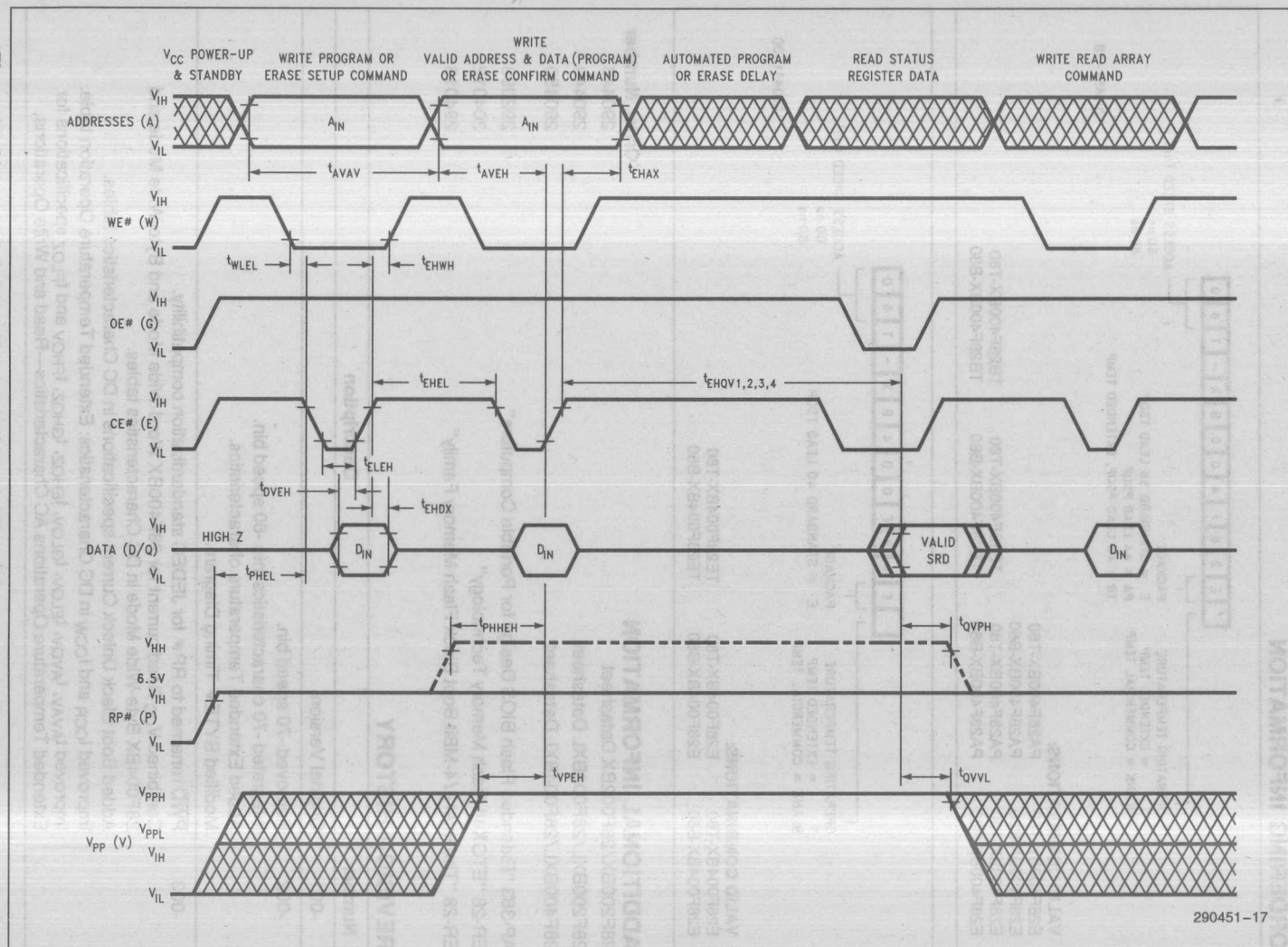
1. Chip-Enable Controlled Writes: Write operations are driven by the valid combination of CE# and WE# in systems where CE# defines the write pulse-width (within a longer WE# timing waveform), all set-up, hold and inactive WE# times should be measured relative to the CE# waveform.

2, 3, 4, 5, 6, 7, 8: Refer to AC Characteristics for WE#-Controlled Write Operations.

9. Read timing characteristics during write and erase operations are the same as during read-only operations. Refer to AC Characteristics during Read Mode.

10. See Standard Test Configuration.

Figure 22. Alternate A.C. Waveforms for Write and Erase Operations (CE#-Controlled Writes)



290451-17

ORDERING INFORMATION

E 2 8 F 4 0 0 B X - T 6 0											
OPERATING TEMPERATURE T = EXTENDED TEMP BLANK = COMMERCIAL TEMP				PACKAGE E = STANDARD 56 LEAD TSOP PA = 44 LEAD PSOP TB = 44 LEAD PSOP, EXTENDED TEMP				ACCESS SPEED (ns) 60 ns 80 ns			
290451-18											
VALID COMBINATIONS:											
E28F400BX-T60			PA28F400BX-T60								
E28F400BX-B60			PA28F400BX-B60								
E28F400BX-T80			PA28F400BX-T80			TE28F400BX-T80			TB28F400BX-T80		
E28F400BX-B80			PA28F400BX-B80			TE28F400BX-B80			TB28F400BX-B80		

E 2 8 F 0 0 4 B X - T 6 0											
OPERATING TEMPERATURE T = EXTENDED TEMP BLANK = COMMERCIAL TEMP				PACKAGE E = STANDARD 40 LEAD TSOP				ACCESS SPEED (ns) 60 ns 80 ns			
290451-30											
VALID COMBINATIONS:											
E28F004BX-T60			E28F004BX-T80			TE28F004BX-T80					
E28F004BX-B60			E28F004BX-B80			TE28F004BX-B80					

ADDITIONAL INFORMATION

	Order Number
28F200BX/28F002BX Datasheet	290448
28F200BXL/28F002BXL Datasheet	290449
28F400BXL/28F004BXL Datasheet	290450
AP-363 "Extended Flash BIOS Design for Portable Computers"	292098
ER-28 "ETOX-III Flash Memory Technology"	204012
ER-29 "The Intel 2/4-MBit Boot Block Flash Memory Family"	294013

REVISION HISTORY

Number	Description
-001	Original Version
-002	Removed -70 speed bin. Integrated -70 characteristics into -60 speed bin. Added Extended Temperature characteristics. Modified BYTE # Timing Diagram.
-003	PWD renamed to RP# for JEDEC standardization compatibility. Combined V _{CC} Read current for 28F400BX Word-Wide Mode and Byte-Wide Mode and 28F004BX Byte-Wide Mode in DC Characteristics tables. Added Boot Block Unlock Current specifications in DC Characteristics tables. Improved I _{CCR} and I _{CCW} in DC Characteristics: Extended Temperature Operation table. Improved t _{AVAV} , t _{AVQV} , t _{ELQV} , t _{GLQV} , t _{EHQZ} , t _{GHQZ} , t _{FHQV} and t _{FLQZ} specifications for Extended Temperature Operations AC Characteristics—Read and Write Operations.



28F400BX-TL/BL, 28F004BX-TL/BL 4-MBIT (256K x 16, 512K x 8) LOW POWER BOOT BLOCK FLASH MEMORY FAMILY

- **Low Voltage Operation for Very Low Power Portable Applications**
 - $V_{CC} = 3.3V \pm 0.3V$
- **x8/x16 Input/Output Architecture**
 - 28F400BX-TL, 28F400BX-BL
 - For High Performance and High Integration 16-bit and 32-bit CPUs
- **x8-only Input/Output Architecture**
 - 28F004BX-TL, 28F004BX-BL
 - For Space Constrained 8-bit Applications
- **Optimized High Density Blocked Architecture**
 - One 16-KB Protected Boot Block
 - Two 8-KB Parameter Blocks
 - One 96-KB Main Block
 - Three 128-KB Main Blocks
 - Top or Bottom Boot Locations
- **Extended Cycling Capability**
 - 10,000 Block Erase Cycles
- **Automated Word/Byte Write and Block Erase**
 - Command User Interface
 - Status Registers
 - Erase Suspend Capability
- **SRAM-Compatible Write Interface**
- **Automatic Power Savings Feature**
 - 0.8 mA typical I_{CC} Active Current in Static Operation
- **Very High-Performance Read**
 - 150 ns Maximum Access Time
 - 65 ns Maximum Output Enable Time
- **Low Power Consumption**
 - 15 mA Typical Active Read Current
- **Reset/Deep Power-Down Input:**
 - 0.2 μA I_{CC} Typical
 - Acts as Reset for Boot Operations
- **Write Protection for Boot Block**
- **Hardware Data Protection Feature**
 - Erase/Write Lockout During Power Transitions
- **Industry Standard Surface Mount Packaging**
 - 28F400BX: JEDEC ROM Compatible
 - 44-Lead PSOP
 - 56-Lead TSOP
 - 28F004BX: 40-Lead TSOP
- **12V Word/Byte Write and Block Erase**
 - $V_{pp} = 12V \pm 5\%$ Standard
- **ETOX III Flash Technology**
 - 3.3V Read

Intel's 4-Mbit Low Power Flash Memory Family is an extension of the Boot Block Architecture which includes block-selective erasure, automated write and erase operations and standard microprocessor interface. The 4-Mbit Low Power Flash Memory Family enhances the Boot Block Architecture by adding more density and blocks, x8/x16 input/output control, very low power, very high speed, an industry standard ROM compatible pinout and surface mount packaging. The 4-Mbit low power flash family opens a new capability for 3V battery-operated portable systems and is an easy upgrade to Intel's 2-Mbit Low Power Boot Block Flash Memory Family.

The Intel 28F400BX-TL/BL are 16-bit wide low power flash memory offerings. These high density flash memories provide user selectable bus operation for either 8-bit or 16-bit applications. The 28F400BX-TL and 28F400BX-BL are 4,194,304-bit non-volatile memories organized as either 524,288 bytes or 262,144 words of information. They are offered in 44-Lead plastic SOP and 56-Lead TSOP packages. The x8/x16 pinout conforms to the industry standard ROM/EPROM pinout. The Intel 28F004BX-TL/BL are 8-bit wide low power flash memories with 4,194,304 bits organized as 524,288 bytes of information. They are offered in a 40-Lead TSOP package, which is ideal for space-constrained portable systems.

These devices use an integrated Command User Interface (CUI) and Write State Machine (WSM) for simplified word/byte write and block erasure. The 28F400BX-TL/28F004BX-TL provide block locations compatible with Intel's Low Voltage MCS-186 family, i386™, i486™ microprocessors. The 28F400BX-BL/28F004BX-BL provide compatibility with Intel's 80960KX and 80960SX families as well as other low voltage embedded microprocessors.

The boot block includes a data protection feature to protect the boot code in critical applications. With a maximum access time of 150 ns, these 4-Mbit low power flash devices are very high performance memories at 3.3V which interface to a wide range of low voltage microprocessors and microcontrollers. A deep power-down mode lowers the total V_{CC} power consumption to 0.66 μW which is critical in handheld battery powered systems such as Handy Cellular Phones. For very high speed applications using a 5V supply, refer to the Intel 28F400BX-T/B, 28F004BX-T/B 4-Mbit Boot Block Flash Memory family datasheet.

Manufactured on Intel's 0.8 micron ETOX III process, the 4-Mbit flash memory family provides world class quality, reliability and cost-effectiveness at the 4 Mbit density level.

1.0 PRODUCT FAMILY OVERVIEW

Throughout this datasheet 28F400BX-L refers to both the 28F400BX-TL and 28F400BX-BL devices and 28F004BX-L refers to both the 28F004BX-TL and 28F004BX-BL devices. The 4-Mbit flash family refers to both the 28F400BX-L and 28F004BX-L products. This datasheet comprises the specifications for four separate products in the 4-Mbit flash family, Section 1 provides an overview of the 4-Mbit flash family including applications, pinouts and pin descriptions. Sections 2 and 3 describe in detail the specific memory organizations for the 28F400BX-L and 28F004BX-L products respectively, Section 4 combines a description of the family's principles of operations. Finally section 5 describes the family's operating specifications.

Product Family

x8/x16 Products	x8-Only Products
28F400BX-TL	28F004BX-TL
28F400BX-BL	28F004BX-BL

1.1 Main Features

The 28F400BX-L/28F004BX-L boot block flash memory family is a very high performance 4-Mbit (4,194,304 bit) memory family organized as either 256 KWords (262,144 words) of 16 bits each or 512 Kbytes (524,288 bytes) of 8 bits each.

Seven Separately Erasable Blocks including a **Hardware-Lockable boot block** (16,384 Bytes), **two parameter blocks** (8,192 Bytes each) and **four main blocks** (1 block of 98,304 Bytes and 3 blocks of 131,072 Bytes) are included on the 4-Mbit family. An erase operation erases one of the main blocks in typically 3.4 seconds and the boot or parameter blocks in typically 2.0 seconds, independent of the remaining blocks. Each block can be independently erased and programmed 10,000 times.

The Boot Block is located at either the top (28F400BX-TL, 28F004BX-TL) or the bottom (28F400BX-BL, 28F004BX-BL) of the address map in order to accommodate different microprocessor protocols for boot code location. The **hardware lockable boot block** provides the most secure code storage. The boot block is intended to store the kernel code required for booting-up a system. When the **RP#** pin is between 11.4V and 12.6V the boot block is unlocked and program and erase operations can be performed. When the **RP#** pin is at or below 4.1V the boot block is locked and program and erase operations to the boot block are ignored.

The 28F400BX-L products are available in the ROM/EPROM compatible pinout and housed in the 44-Lead PSOP (Plastic Small Outline) package and the 56-Lead TSOP (Thin Small Outline, 1.2mm thick) package as shown in Figures 3 and 4. The 28F004BX-L products are available in the 40-Lead TSOP (1.2mm thick) package as shown in Figure 5.

The **Command User Interface (CUI)** serves as the interface between the microprocessor or microcontroller and the internal operation of the 28F400BX-L and 28F004BX-L flash memory products.

Program and Erase Automation allow program and erase operations to be executed using a two-write command sequence to the CUI. The internal Write State Machine (WSM) automatically executes the algorithms and timings necessary for program and erase operations, including verifications, thereby unburdening the microprocessor or microcontroller. Writing of memory data is performed in word or byte increments for the 28F400BX-L family and in byte increments for the 28F004BX-L family typically within 11 μ s.

The **Status Register (SR)** indicates the status of the WSM and whether the WSM successfully completed the desired program or erase operation.

Maximum Access Time of **150 ns (TACC)** is achieved over the commercial temperature range (0°C to +70°C), V_{CC} supply voltage range (3.0V to 3.6V, 4.5V to 5.5V) and 50 pF output load.

I_{pp} Program current is 40 mA for x16 operation and 30 mA for x8 operation. I_{pp} Erase current is 30 mA maximum. V_{pp} erase and programming voltage is 11.4V to 12.6V (V_{pp} = 12V \pm 5%) under all operating conditions.

Typical I_{CC} Active Current of 15 mA is achieved for the x16 products and the x8 products.

The 4-Mbit flash family is also designed with an Automatic Power Savings (APS) feature to minimize system battery current drain and allow for very low power designs. Once the device is accessed to read the array data, APS mode will immediately put the memory in static mode of operation where I_{CC} active current is typically 0.8 mA until the next read is initiated.

When the **CE#** and **RP#** pins are at V_{CC} and the **BYTE#** pin (28F400BX-L-only) is at either V_{CC} or GND the **CMOS Standby** mode is enabled where I_{CC} is typically 45 μ A.

A **Deep Power-Down Mode** is enabled when the PWD pin is at ground minimizing power consumption and providing write protection during power-up conditions. **I_{CC} current** during deep power-down mode is **0.20 μ A typical**. An initial maximum access time or Reset Time of 700 ns is required from RP# switching until outputs are valid. Equivalently, the device has a maximum wake-up time of 580 ns until writes to the Command User Interface are recognized. When RP# is at ground the WSM is reset, the Status Register is cleared and the entire device is protected from being written to. This feature prevents data corruption and protects the code stored in the device during system reset. The system Reset pin can be tied to RP# to reset the memory to normal read mode upon activation of the Reset pin. When the CPU enters reset mode, it expects to read the contents of a memory location. Furthermore, with on-chip program/erase automation in the 4-Mbit family and the RP# functionality for data protection, when the CPU is reset and even if a program or erase command is issued, the device will not recognize any operation until RP# returns to its normal state.

For the 28F400BX-L, **Byte-wide or Word-wide Input/Output Control** is possible by controlling the BYTE# pin. When the BYTE# pin is at a logic low the device is in the byte-wide mode (x8) and data is read and written through DQ[0:7]. During the byte-wide mode, DQ[8:14] are tri-stated and DQ15/A-1 becomes the lowest order address pin. When the BYTE# pin is at a logic high the device is in the word-wide mode (x16) and data is read and written through DQ[0:15].

1.2 Applications

The 4-Mbit low power boot block flash memory family combines high density, very low power, high performance, cost-effective flash memories with blocking and hardware protection capabilities. Its flexibility and versatility will reduce costs throughout the product life cycle. Flash memory is ideal for Just-In-Time production flow, reducing system inventory and costs, and eliminating component handling during the production phase. During the product life cycle, when code updates or feature enhancements become necessary, flash memory will reduce the update costs by allowing either a user-performed code change via floppy disk or a remote code change via

a serial link. The 4-Mbit flash family provides full function, blocked flash memories suitable for a wide range of applications. These applications include **Extended PC BIOS and ROM-able** applications storage, **Handy Digital Cellular Phone** program and data storage and various other low power embedded applications where both program and data storage are required.

Portable systems such as Notebook/Palmtop computers, are ideal applications for the 4-Mbit low power flash products. Portable and handheld personal computer applications are becoming more complex with the addition of power management software to take advantage of the latest microprocessor technology, the availability of ROM-based application software, pen tablet code for electronic hand writing, and diagnostic code. Figure 1 shows an example of a 28F400BX-TL application.

This increase in software sophistication augments the probability that a code update will be required after the Notebook is shipped. The 4-Mbit flash products provide an inexpensive update solution for the notebook and handheld personal computers while extending their product lifetime. Furthermore, the 4-Mbit flash products' deep power-down mode provides added flexibility for these battery-operated portable designs which require operation at very low power levels.

The 4-Mbit low power flash products also provide excellent design solutions for Handy Digital Cellular Phone applications requiring low voltage supply, high performance, high density storage capability coupled with modular software designs, and a small form factor package (x8-only bus). The 4-Mbit's blocking scheme allows for an easy segmentation of the embedded code with; 16-Kbytes of Hardware-Protected Boot code, 4 Main Blocks of program code and 2 Parameter Blocks of 8-Kbytes each for frequently updatable data storage and diagnostic messages (e.g., phone numbers, authorization codes). Figure 2 is an example of such an application with the 28F004BX-TL.

These are a few actual examples of the wide range of applications for the 4-Mbit low power Boot Block flash memory family which enable system designers achieve the best possible product design. Only your imagination limits the applicability of such a versatile low power product family.

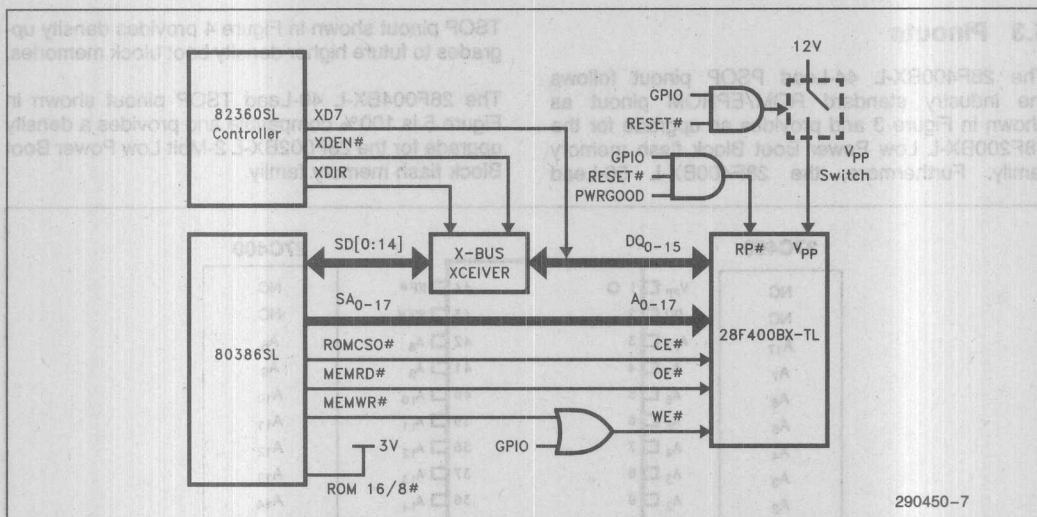


Figure 1. 28F400BX-L Interface to INTEL386SL 3.3V Microprocessor Superset

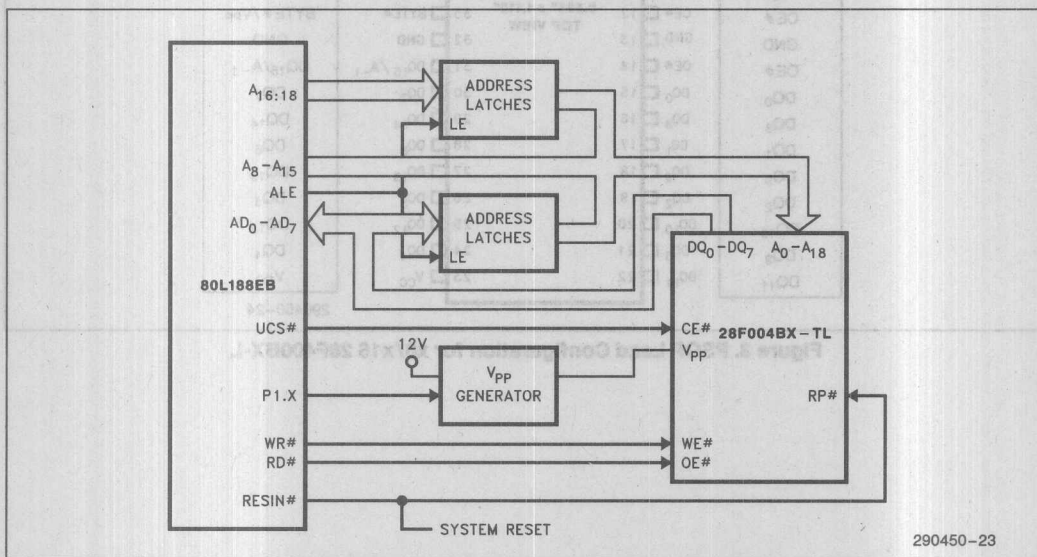


Figure 2. 28F004BX-L Interface to INTEL 80L188EB Low Voltage 8-bit Embedded Microprocessor

1.3 Pinouts

The 28F400BX-L 44-Lead PSOP pinout follows the industry standard ROM/EPROM pinout as shown in Figure 3 and provides an upgrade for the 28F200BX-L Low Power Boot Block flash memory family. Furthermore, the 28F400BX-L 56-Lead

TSOP pinout shown in Figure 4 provides density upgrades to future higher density boot block memories.

The 28F004BX-L 40-Lead TSOP pinout shown in Figure 5 is 100% compatible and provides a density upgrade for the 28F002BX-L 2-Mbit Low Power Boot Block flash memory family.

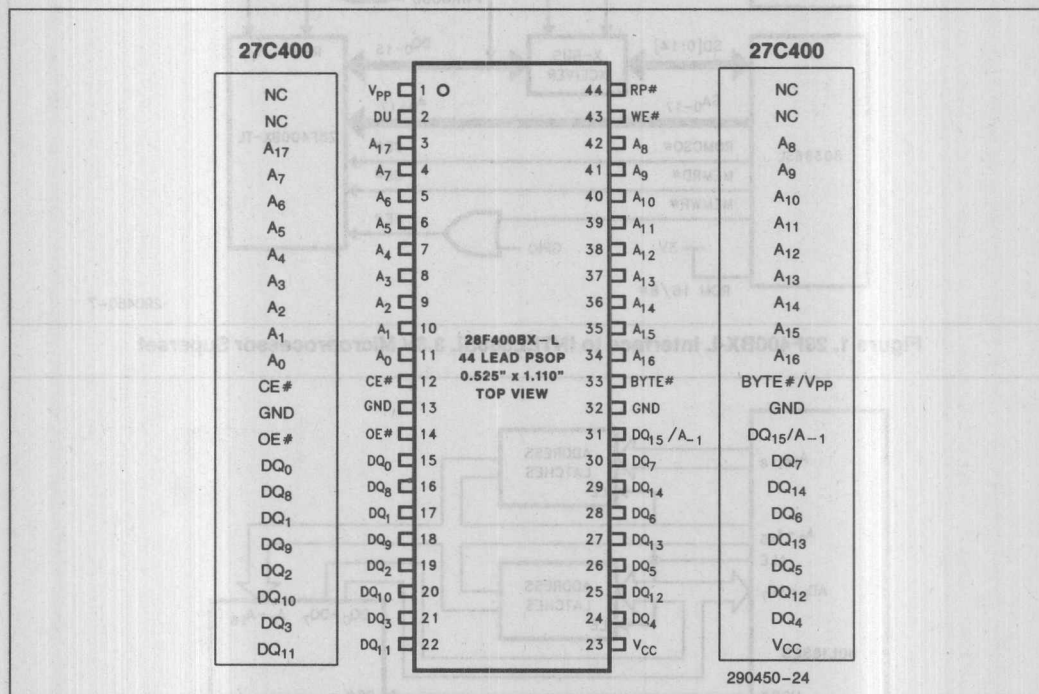
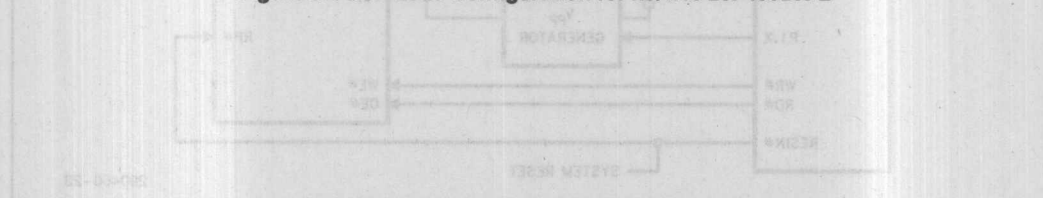


Figure 3. PSOP Lead Configuration for x8/x16 28F400BX-L



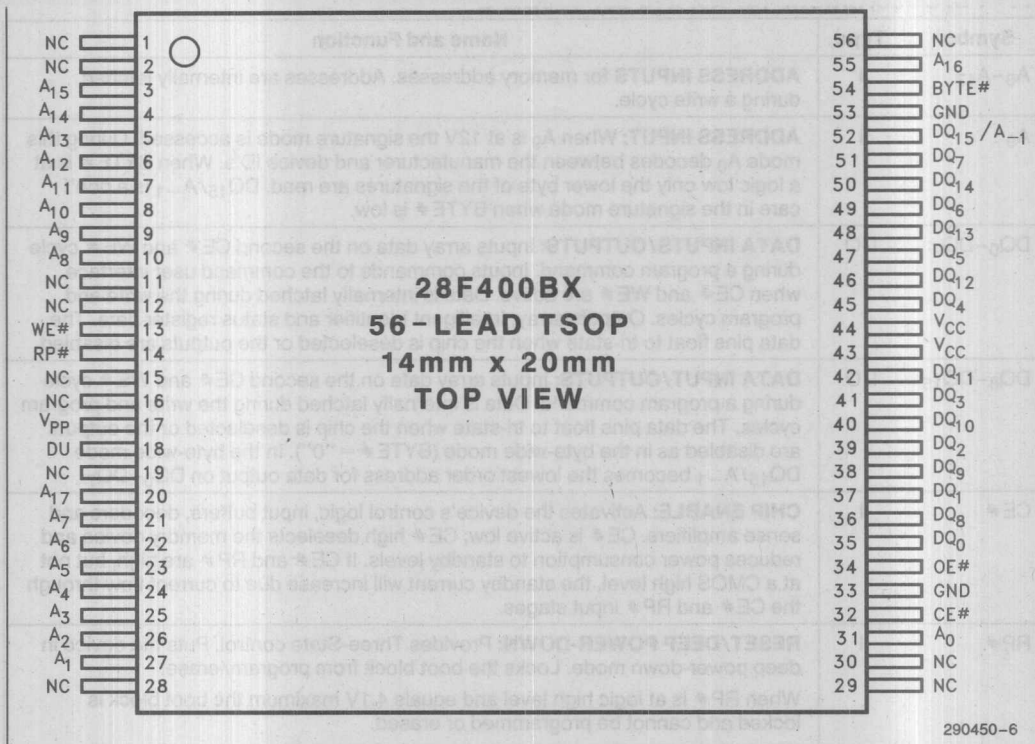


Figure 4. TSOP Lead Configuration for x8 28F400BX-L

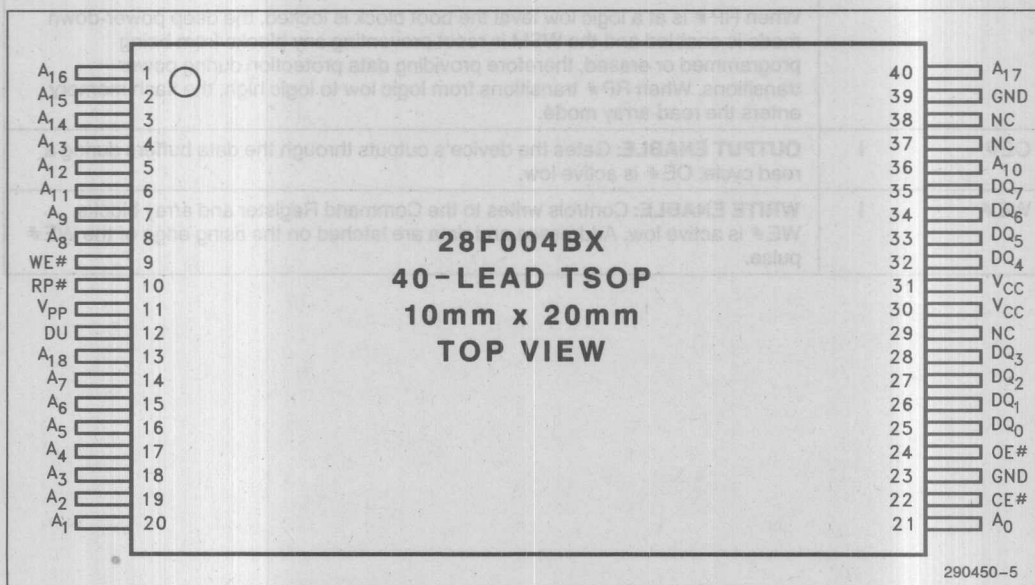


Figure 5. TSOP Lead Configuration for x8 28F004BX-L

1.4 Pin Descriptions for x8/x16 28F400BX-L

Symbol	Type	Name and Function
A ₀ –A ₁₇	I	ADDRESS INPUTS for memory addresses. Addresses are internally latched during a write cycle.
A ₉	I	ADDRESS INPUT: When A ₉ is at 12V the signature mode is accessed. During this mode A ₀ decodes between the manufacturer and device ID's. When BYTE # is at a logic low only the lower byte of the signatures are read. DQ ₁₅ /A _{–1} is a don't care in the signature mode when BYTE # is low.
DQ ₀ –DQ ₇	I/O	DATA INPUTS/OUTPUTS: Inputs array data on the second CE # and WE # cycle during a program command. Inputs commands to the command user interface when CE # and WE # are active. Data is internally latched during the write and program cycles. Outputs array, Intelligent Identifier and status register data. The data pins float to tri-state when the chip is deselected or the outputs are disabled.
DQ ₈ –DQ ₁₅	I/O	DATA INPUT/OUTPUTS: Inputs array data on the second CE # and WE # cycle during a program command. Data is internally latched during the write and program cycles. The data pins float to tri-state when the chip is deselected or the outputs are disabled as in the byte-wide mode (BYTE # = "0"). In the byte-wide mode DQ ₁₅ /A _{–1} becomes the lowest order address for data output on DQ ₀ –DQ ₇ .
CE #	I	CHIP ENABLE: Activates the device's control logic, input buffers, decoders and sense amplifiers. CE # is active low; CE # high deselects the memory device and reduces power consumption to standby levels. If CE # and RP # are high, but not at a CMOS high level, the standby current will increase due to current flow through the CE # and RP # input stages.
RP #	I	RESET/DEEP POWER-DOWN: Provides Three-State control. Puts the device in deep power-down mode. Locks the boot block from program/erase. When RP # is at logic high level and equals 4.1V maximum the boot block is locked and cannot be programmed or erased. When RP # = 11.4V minimum the boot block is unlocked and can be programmed or erased. When RP # is at a logic low level the boot block is locked, the deep power-down mode is enabled and the WSM is reset preventing any blocks from being programmed or erased, therefore providing data protection during power transitions. When RP # transitions from logic low to logic high, the flash memory enters the read-array mode.
OE #	I	OUTPUT ENABLE: Gates the device's outputs through the data buffers during a read cycle. OE # is active low.
WE #	I	WRITE ENABLE: Controls writes to the Command Register and array blocks. WE # is active low. Addresses and data are latched on the rising edge of the WE # pulse.

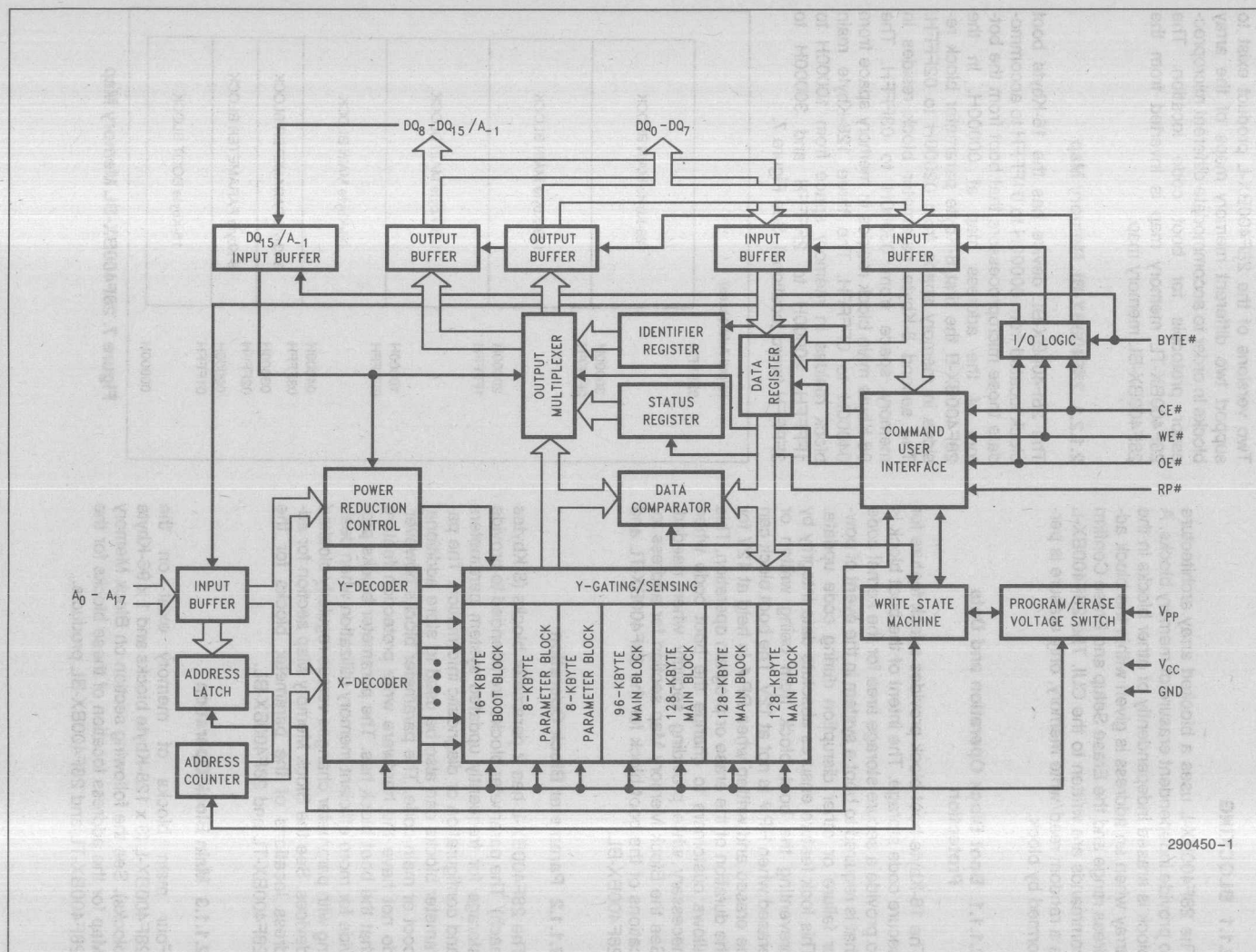
1.4 Pin Descriptions for x8/x16 28F400BX-L (Continued)

Symbol	Type	Name and Function
BYTE #	I	BYTE # ENABLE: Controls whether the device operates in the byte-wide mode (x8) or the word-wide mode (x16). BYTE # = "0" enables the byte-wide mode, where data is read and programmed on DQ ₀ –DQ ₇ and DQ ₁₅ /A _{–1} becomes the lowest order address that decodes between the upper and lower byte. DQ ₈ –DQ ₁₄ are tri-stated during the byte-wide mode. BYTE # = "1" enables the word-wide mode where data is read and programmed on DQ ₀ –DQ ₁₅ .
V _{PP}		PROGRAM/ERASE POWER SUPPLY: For erasing memory array blocks or programming data in each block. Note: V _{PP} < V _{PPLMAX} memory contents cannot be altered.
V _{CC}		DEVICE POWER SUPPLY (3.3V ± 0.3, 5V ± 10%)
GND		GROUND: For all internal circuitry.
NC		NO CONNECT: Pin may be driven or left floating.
DU		DON'T USE PIN: Pin should not be connected to anything.

1.5 Pin Descriptions for x8 28F004BX-L

Symbol	Type	Name and Function
A ₀ -A ₁₈	I	ADDRESS INPUTS for memory addresses. Addresses are internally latched during a write cycle.
A ₉	I	ADDRESS INPUT: When A ₉ is at 12V the signature mode is accessed. During this mode A ₀ decodes between the manufacturer and device ID's.
DQ ₀ -DQ ₇	I/O	DATA INPUTS/OUTPUTS: Inputs array data on the second CE # and WE # cycle during a program command. Inputs commands to the command user interface when CE # and WE # are active. Data is internally latched during the write and program cycles. Outputs array, Intelligent Identifier and status register data. The data pins float to tri-state when the chip is deselected or the outputs are disabled.
CE #	I	CHIP ENABLE: Activates the device's control logic, input buffers, decoders and sense amplifiers. CE # is active low; CE # high deselects the memory device and reduces power consumption to standby levels.
RP #	I	<p>RESET/DEEP POWER-DOWN: Provides Three-State control. Puts the device in deep power-down mode. Locks the Boot Block from program/erase.</p> <p>When RP # is at logic high level and equals 4.1V maximum the Boot Block is locked and cannot be programmed or erased.</p> <p>When RP # = 11.4V minimum the Boot Block is unlocked and can be programmed or erased.</p> <p>When RP # is at a logic low level the Boot Block is locked, the deep power-down mode is enabled and the WSM is reset preventing any blocks from being programmed or erased, therefore providing data protection during power transitions. When RP # transitions from logic low to logic high, the flash memory enters the read-array mode.</p>
OE #	I	OUTPUT ENABLE: Gates the device's outputs through the data buffers during a read cycle. OE # is active low.
WE #	I	WRITE ENABLE: Controls writes to the Command Register and array blocks. WE # is active low. Addresses and data are latched on the rising edge of the WE # pulse.
V _{PP}		<p>PROGRAM/ERASE POWER SUPPLY: For erasing memory array blocks or programming data in each block.</p> <p>Note: V_{PP} < V_{PPLMAX} memory contents cannot be altered.</p>
V _{CC}		DEVICE POWER SUPPLY (3.3V ± 0.3V, 5V ± 10%)
GND		GROUND: For all internal circuitry.
NC		NO CONNECT: Pin may be driven or left floating.
DU		DON'T USE PIN: Pin should not be connected to anything.

Figure 6. 28F400BX-L Word/Byte-Wide Block Diagram



290450-1

2.1 28F400BX-L Memory Organization

2.1.1 BLOCKING

The 28F400BX-L uses a blocked array architecture to provide independent erasure of memory blocks. A block is erased independently of other blocks in the array when an address is given within the block address range and the Erase Setup and Erase Confirm commands are written to the CUI. The 28F400BX-L is a random read/write memory, only erasure is performed by block.

2.1.1.1 Boot Block Operation and Data Protection

The 16-Kbyte boot block provides a lock feature for secure code storage. The intent of the boot block is to provide a secure storage area for the kernel code that is required to boot a system in the event of power failure or other disruption during code update. This lock feature ensures absolute data integrity by preventing the boot block from being written or erased when RP# is not at 12V. The boot block can be erased and written when RP# is held at 12V for the duration of the erase or program operation. This allows customers to change the boot code when necessary while providing security when needed. See the Block Memory Map section for address locations of the boot block for the 28F400BX-TL and 28F400BX-BL.

2.1.1.2 Parameter Block Operation

The 28F400BX-L has 2 parameter blocks (8 Kbytes each). The parameter blocks are intended to provide storage for frequently updated system parameters and configuration or diagnostic information. The parameter blocks can also be used to store additional boot or main code. The parameter blocks however, do not have the hardware write protection feature that the boot block has. The parameter blocks provide for more efficient memory utilization when dealing with parameter changes versus regularly blocked devices. See the Block Memory Map section for address locations of the parameter blocks for the 28F400BX-TL and 28F400BX-BL.

2.1.1.3 Main Block Operation

Four main blocks of memory exist on the 28F400BX-L (3 x 128-Kbyte blocks and 1 x 96-Kbyte blocks). See the following section on Block Memory Map for the address location of these blocks for the 28F400BX-TL and 28F400BX-BL products.

2.1.2 BLOCK MEMORY MAP

Two versions of the 28F400BX-L product exist to support two different memory maps of the array blocks in order to accommodate different microprocessor protocols for boot code location. The 28F400BX-TL memory map is inverted from the 28F400BX-BL memory map.

2.1.2.1 28F400BX-BL Memory Map

The 28F400BX-BL device has the 16-Kbyte boot block located from 00000H to 01FFFFH to accommodate those microprocessors that boot from the bottom of the address map at 00000H. In the 28F400BX-B the first 8-Kbyte parameter block resides in memory space from 02000H to 02FFFFH. The second 8-Kbyte parameter block resides in memory space from 03000H to 03FFFFH. The 96-Kbyte main block resides in memory space from 04000H to 0FFFFH. The three 128-Kbyte main block resides in memory space from 10000H to 1FFFFH, 20000H to 2FFFFH and 30000H to 3FFFFH (word locations). See Figure 7.

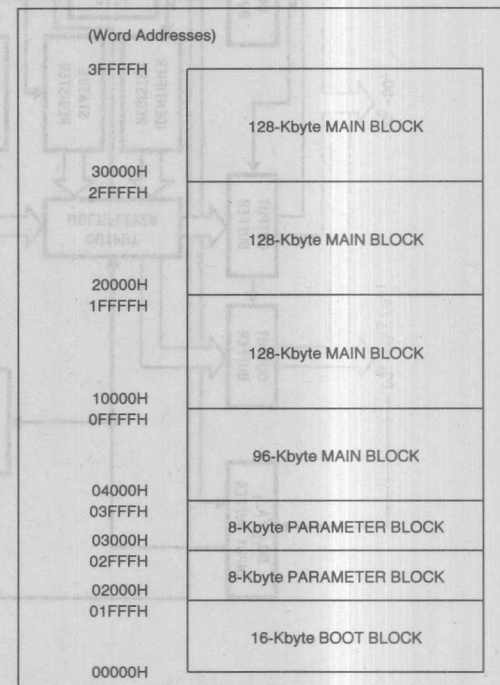


Figure 7. 28F400BX-BL Memory Map

2.1.2.2 28F400BX-TL Memory Map

The 28F400BX-TL device has the 16-Kbyte boot block located from 3E000H to 3FFFFH to accommodate those microprocessors that boot from the top of the address map. In the 28F400BX-T the first 8-Kbyte parameter block resides in memory space from 3D000H to 3DFFFH. The second 8-Kbyte parameter block resides in memory space from 3C000H to 3CFFFH. The 96-Kbyte main block resides in memory space from 30000H to 3BFFFH. The three 128-Kbyte main blocks reside in memory space from 20000H to 2FFFFH, 10000H to 1FFFFH and 00000H to 0FFFFH as shown in Figure 8.

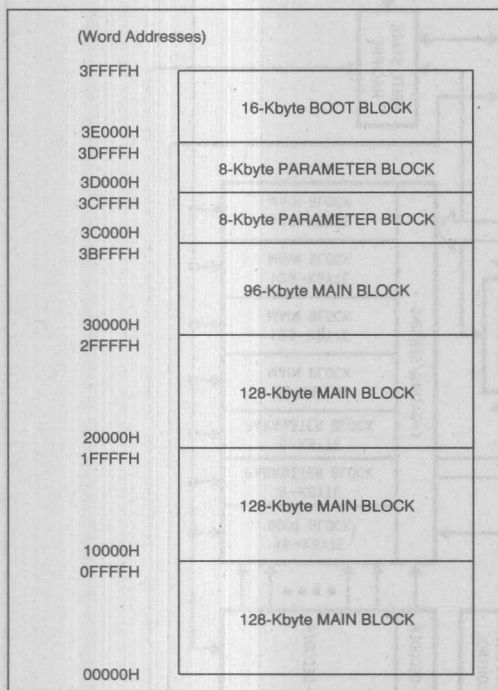


Figure 8. 28F400BX-TL Memory Map

290450-3



4-62

PRELIMINARY

3.1 28F004BX-L Memory Organization

3.1.1 BLOCKING

The 28F004BX-L uses a blocked array architecture to provide independent erasure of memory blocks. A block is erased independently of other blocks in the array when an address is given within the block address range and the Erase Setup and Erase Confirm commands are written to the CUI. The 28F004BX-L is a random read/write memory, only erasure is performed by block.

3.1.1.1 Boot Block Operation and Data Protection

The 16-Kbyte boot block provides a lock feature for secure code storage. The intent of the boot block is to provide a secure storage area for the kernel code that is required to boot a system in the event of power failure or other disruption during code update. This lock feature ensures absolute data integrity by preventing the boot block from being programmed or erased when RP# is not at 12V. The boot block can be erased and programmed when RP# is held at 12V for the duration of the erase or program operation. This allows customers to change the boot code when necessary while still providing security when needed. See the Block Memory Map section for address locations of the boot block for the 28F004BX-TL and 28F004BX-BL.

3.1.1.2 Parameter Block Operation

The 28F004BX-L has 2 parameter blocks (8 Kbytes each). The parameter blocks are intended to provide storage for frequently updated system parameters and configuration or diagnostic information. The parameter blocks can also be used to store additional boot or main code. The parameter blocks however, do not have the hardware write protection feature that the boot block has. Parameter blocks provide for more efficient memory utilization when dealing with small parameter changes versus regularly blocked devices. See the Block Memory Map section for address locations of the parameter blocks for the 28F004BX-TL and 28F004BX-BL.

3.1.1.3 Main Block Operation

Two main blocks of memory exist on the 28F004BX-L (3 x 128-Kbyte blocks and 1 x 96-Kbyte blocks). See the following section on Block Memory Map for the address location of these blocks for the 28F004BX-TL and 28F004BX-BL.

3.1.2 BLOCK MEMORY MAP

Two versions of the 28F004BX-L product exist to support two different memory maps of the array blocks in order to accommodate different microprocessor protocols for boot code location. The 28F004BX-TL memory map is inverted from the 28F004BX-BL memory map.

3.1.2.1 28F004BX-BL Memory Map

The 28F004BX-BL device has the 16-Kbyte boot block located from 00000H to 03FFFFH to accommodate those microprocessors that boot from the bottom of the address map at 00000H. In the 28F004BX-BL the first 8-Kbyte parameter block resides in memory from 04000H to 05FFFFH. The second 8-Kbyte parameter block resides in memory space from 06000H to 07FFFFH. The 96-Kbyte main block resides in memory space from 08000H to 1FFFFH. The three 128-Kbyte main blocks reside in memory space from 20000H to 3FFFFH, 40000H to 5FFFFH and 60000H to 7FFFFH. See Figure 10.

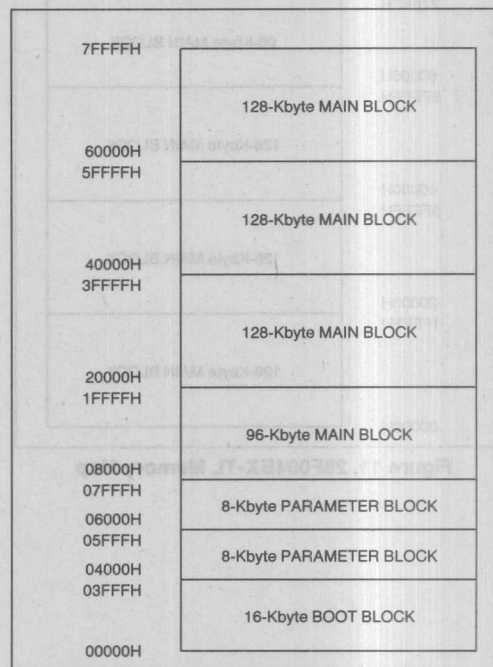


Figure 10. 28F004BX-BL Memory Map

3.1.2.2 28F004BX-TL Memory Map

The 28F004BX-TL device has the 16-Kbyte boot block located from 7C000H to 7FFFFH to accommodate those microprocessors that boot from the top of the address map. In the 28F004BX-TL the first 8-Kbyte parameter block resides in memory space from 7A000H to 7BFFFFH. The second 8-Kbyte parameter block resides in memory space from 78000H to 79FFFFH. The 96-Kbyte main block resides in memory space from 60000H to 77FFFFH. The three 128-Kbyte main blocks reside in memory space from 40000H to 5FFFFH, 20000H to 3FFFFH and 00000H to 1FFFFH.

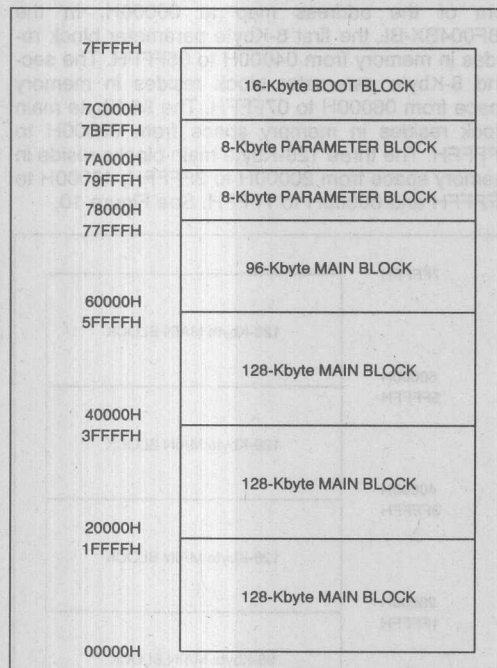


Figure 11. 28F004BX-TL Memory Map

4.0 PRODUCT FAMILY PRINCIPLES OF OPERATION

Flash memory augments EPROM functionality with in-circuit electrical write and erase. The 4-Mbit flash memory family utilizes a Command User Interface (CUI) and internally generated and timed algorithms to simplify write and erase operations.

The CUI allows for fixed power supplies during erase and programming, and maximum EPROM compatibility.

In the absence of high voltage on the V_{PP} pin, the 4-Mbit flash family will only successfully execute the following commands: Read Array, Read Status Register, Clear Status Register and Intelligent Identifier mode. The device provides standard EPROM read, standby and output disable operations. Manufacturer Identification and Device Identification data can be accessed through the CUI or through the standard EPROM A9 high voltage access (V_{ID}) (for PROM programmer equipment).

The same EPROM read, standby and output disable functions are available when high voltage is applied to the V_{PP} pin. In addition, high voltage on V_{PP} allows write and erase of the device. All functions associated with altering memory contents: write and erase, Intelligent Identifier read and Read Status are accessed via the CUI.

The purpose of the Write State Machine (WSM) is to completely automate the write and erasure of the device. The WSM will begin operation upon receipt of a signal from the CUI and will report status back through a Status Register. The CUI will handle the WE# interface to the data and address latches, as well as system software requests for status while the WSM is in operation.

4.1 28F400BX-L Bus Operations

Flash memory reads, erases and writes in-system via the local CPU. All bus cycles to or from the flash memory conform to standard microprocessor bus cycles.

Table 1. Bus Operations for WORD-WIDE Mode (BYTE# = V_{IH})

Mode	Notes	RP#	CE#	OE#	WE#	A ₉	A ₀	V _{PP}	DQ ₀₋₁₅
Read	1, 2, 3	V _{IH}	V _{IL}	V _{IL}	V _{IH}	X	X	X	D _{OUT}
Output Disable		V _{IH}	V _{IL}	V _{IH}	V _{IH}	X	X	X	High Z
Standby		V _{IH}	V _{IH}	X	X	X	X	X	High Z
Deep Power-Down	9	V _{IL}	X	X	X	X	X	X	High Z
Intelligent Identifier (Mfr)	4	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IL}	X	0089H
Intelligent Identifier (Device)	4, 5, 10	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IH}	X	4470H 4471H
Write	6, 7, 8	V _{IH}	V _{IL}	V _{IH}	V _{IL}	X	X	X	D _{IN}

Table 2. Bus Operations for BYTE-WIDE Mode (BYTE# = V_{IL})

Mode	Notes	RP#	CE#	OE#	WE#	A ₉	A ₀	A ₋₁	V _{PP}	DQ ₀₋₇	DQ ₈₋₁₄
Read	1, 2, 3	V _{IH}	V _{IL}	V _{IL}	V _{IH}	X	X	X	X	D _{OUT}	High Z
Output Disable		V _{IH}	V _{IL}	V _{IH}	V _{IH}	X	X	X	X	High Z	High Z
Standby		V _{IH}	V _{IH}	X	X	X	X	X	X	High Z	High Z
Deep Power-Down		V _{IL}	X	X	X	X	X	X	X	High Z	High Z
Intelligent Identifier (Mfr)	4	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IL}	X	X	89H	High Z
Intelligent Identifier (Device)	4, 5, 10	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IH}	X	X	70H 71H	High Z
Write	6, 7, 8	V _{IH}	V _{IL}	V _{IH}	V _{IL}	X	X	X	X	D _{IN}	High Z

NOTES:

1. Refer to DC Characteristics.
2. X can be V_{IL} or V_{IH} for control pins and addresses, V_{DDL} or V_{DDH} for V_{PP}.
3. See DC Characteristics for V_{DDL}, V_{DDH}, V_{HH}, V_{ID} voltages.
4. Manufacturer and Device codes may also be accessed via a CUI write sequence. A₁-A₁₇ = V_{IL}.
5. Device ID = 4470H for 28F400BX-TL and 4471H for 28F400BX-BL.
6. Refer to Table 4 for valid D_{IN} during a write operation.
7. Command writes for Block Erase or Word/Byte Write are only executed when V_{PP} = V_{DDH}.
8. To write or erase the boot block, hold RP# at V_{IH}.
9. RP# must be at GND ± 0.2V to meet the 1.2 μA maximum deep power-down current.
10. The device ID codes are identical to those of the 28F400BX 5V version.

4.2 28F004BX-L Bus Operations

Table 3. Bus Operations

Mode	Notes	RP #	CE #	OE #	WE #	A ₉	A ₀	V _{PP}	DQ ₀₋₇
Read	1, 2, 3	V _{IH}	V _{IL}	V _{IL}	V _{IH}	X	X	X	D _{OUT}
Output Disable		V _{IH}	V _{IL}	V _{IH}	V _{IH}	X	X	X	High Z
Standby		V _{IH}	V _{IH}	X	X	X	X	X	High Z
Deep Power-Down	9	V _{IL}	X	X	X	X	X	X	High Z
Intelligent Identifier (Mfr)	4	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IL}	X	89H
Intelligent Identifier (Device)	4, 5, 10	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IH}	X	78H 79H
Write	6, 7, 8	V _{IH}	V _{IL}	V _{IH}	V _{IL}	X	X	X	D _{IN}

NOTES:

1. Refer to DC Characteristics.
2. X can be V_{IL} or V_{IH} for control pins and addresses, V_{PPL} or V_{PPH} for V_{PP}.
3. See DC Characteristics for V_{PPL}, V_{PPH}, V_{HH}, V_{ID} voltages.
4. Manufacturer and Device codes may also be accessed via a CUI write sequence. A₁-A₈, A₁₀-A₁₈ = V_{IL}.
5. Device ID = 78H for 28F004BX-TL and 79H for 28F004BX-BL.
6. Refer to Table 4 for valid D_{IN} during a write operation.
7. Command writes for Block erase or byte program are only executed when V_{PP} = V_{PPH}.
8. Program or erase the Boot block by holding RP# at V_{HH}.
9. RP# must be at GND ±0.2V to meet the 1.2 µA maximum deep power-down current.
10. The device ID codes are identical to those of the 28F004BX 5V version.

4.3 Read Operations

The 4-Mbit flash family has three user read modes; Array, Intelligent Identifier, and Status Register. Status Register read mode will be discussed in detail in the "Write Operations" section.

During power-up conditions (V_{CC} supply ramping), it takes a maximum of 700 ns from V_{CC} at 3.0V minimum to obtain valid data on the outputs.

4.3.1 READ ARRAY

If the memory is not in the Read Array mode, it is necessary to write the appropriate read mode command to the CUI. The 4-Mbit flash family has three control functions, all of which must be logically active, to obtain data at the outputs. Chip-Enable CE# is the device selection control. Power-Down RP# is the device power control. Output-Enable OE# is the DATA INPUT/OUTPUT (DQ[0:15] or DQ[0:7]) direction control and when active is used to drive data from the selected memory onto the I/O bus.

4.3.1.1 Output Control

With OE# at logic-high level (V_{IH}), the output from the device is disabled and data input/output pins (DQ[0:15] or DQ[0:7]) are tri-stated. Data input is then controlled by WE#.

4.3.1.2 Input Control

With WE# at logic-high level (V_{IH}), input to the device is disabled. Data Input/Output pins (DQ[0:15] or DQ[0:7]) are controlled by OE#.

4.3.2 INTELLIGENT IDENTIFIERS

28F400BX-L PRODUCTS

The manufacturer and device codes are read via the CUI or by taking the A₉ pin to 12V. Writing 90H to the CUI places the device into Intelligent Identifier read mode. A read of location 00000H outputs the manufacturer's identification code, 0089H, and location 00001H outputs the device code; 4470H for 28F400BX-TL, 4471H for 28F4001BX-BL. When BYTE# is at a logic low only the lower byte of the above signatures is read and DQ₁₅/A₋₁ is a "don't care" during Intelligent Identifier mode. A read array command must be written to the CUI to return to the read array mode.

28F004BX-L PRODUCTS

The manufacturer and device codes are also read via the CUI or by taking the A9 pin to 12V. Writing 90H to the CUI places the device into Intelligent Identifier read mode. A read of location 00000H outputs the manufacturer's identification code, 89H, and location 00001H outputs the device code; 78H for 28F004BX-TL, 79H for 28F004BX-BL.

4.4 Write Operations

Commands are written to the CUI using standard microprocessor write timings. The CUI serves as the interface between the microprocessor and the internal chip operation. The CUI can decipher Read Array, Read Intelligent Identifier, Read Status Register, Clear Status Register, Erase and Program commands. In the event of a read command, the CUI simply points the read path at either the array, the Intelligent Identifier, or the status register depending on the specific read command given. For a program or erase cycle, the CUI informs the write state machine that a write or erase has been requested. During a program cycle, the Write State Machine will control the program sequences and the CUI will only respond to status reads. During an erase cycle, the CUI will respond to status reads and erase suspend. After the Write State Machine has completed its task, it will allow the CUI to respond to its full command set. The CUI will stay in the current command state until the microprocessor issues another command.

The CUI will successfully initiate an erase or write operation only when V_{pp} is within its voltage range. Depending upon the application, the system designer may choose to make the V_{pp} power supply switchable, available only when memory updates are desired. The system designer can also choose to "hard-wire" V_{pp} to 12V. The 4-Mbit flash memory family is designed to accommodate either design practice. It is recommended that RP# be tied to logical Reset for data protection during unstable CPU reset function as described in the "Product Family Overview" section.

4.4.1 BOOT BLOCK WRITE OPERATIONS

In the case of Boot Block modifications (write and erase), RP# is set to $V_{HH} = 12V$ typically, in addition to V_{pp} at high voltage.

However, if RP# is not at V_{HH} when a program or erase operation of the boot block is attempted, the corresponding status register bit (Bit 4 for Program and Bit 5 for Erase, refer to Table 5 for Status Register Definitions) is set to indicate the failure to complete the operation.

4.4.2 COMMAND USER INTERFACE (CUI)

The Command User Interface (CUI) serves as the interface to the microprocessor. The CUI points the read/write path to the appropriate circuit block as described in the previous section. After the WSM has completed its task, it will set the WSM Status bit to a "1", which will also allow the CUI to respond to its full command set. Note that after the WSM has returned control to the CUI, the CUI will remain in its current state.

4.4.2.1 Command Set

Command Codes	Device Mode
00	Invalid/Reserved
10	Alternate Program Setup
20	Erase Setup
40	Program Setup
50	Clear Status Register
70	Read Status Register
90	Intelligent Identifier
B0	Erase Suspend
D0	Erase Resume/Erase Confirm
FF	Read Array

4.4.2.2 Command Function Descriptions

Device operations are selected by writing specific commands into the CUI. Table 4 below defines the 4-Mbit flash memory family commands.

Table 4. Command Definitions

Command	Bus Cycles Req'd	Notes 8	First Bus Cycle			Second Bus Cycle		
			Operation	Address	Data	Operation	Address	Data
Read Array	1	1	Write	X	FFH			
Intelligent Identifier	3	2, 4	Write	X	90H	Read	IA	IID
Read Status Register	2	3	Write	X	70H	Read	X	SRD
Clear Status Register	1		Write	X	50H			
Erase Setup/Erase Confirm	2	5	Write	BA	20H	Write	BA	D0H
Word/Byte Write Setup/Write	2	6, 7	Write	WA	40H	Write	WA	WD
Erase Suspend/Erase Resume	2		Write	X	B0H	Write	X	D0H
Alternate Word/Byte Write Setup/Write	2	WD	2, 3, 7	Write	WA	10H	Write	WA

NOTES:

1. Bus operations are defined in Tables 1, 2, 3.
 2. IA = Identifier Address: 00H for manufacturer code, 01H for device code.
 3. SRD = Data read from Status Register.
 4. IID = Intelligent Identifier Data.
- Following the Intelligent Identifier command, two read operations access manufacturer and device codes.
5. BA = Address within the block being erased.
 6. WA = Address to be written.
 7. WD = Data to be written at location WA.
 8. Either 40H or 10H commands is valid.
 9. When writing commands to the device the upper data bus [DQ₈-DQ₁₅] = X (28F400BX-L-only) which is either V_{CC} or V_{SS} to avoid burning additional current.

Invalid/Reserved

These are unassigned commands. It is not recommended that the customer use any command other than the valid commands specified above. Intel reserves the right to redefine these codes for future functions.

Read Array (FFH)

This single write command points the read path at the array. If the host CPU performs a CE#/OE# controlled read immediately following a two-write sequence that started the WSM, then the device will output status register contents. If the Read Array command is given after Erase Setup the device is reset to read the array. A two Read Array command sequence (FFH) is required to reset to Read Array after Program Setup.

Intelligent Identifier (90H)

After this command is executed, the CUI points the output path to the Intelligent Identifier circuits. Only Intelligent Identifier values at addresses 0 and 1 can be read (only address A0 is used in this mode, all other address inputs are ignored).

Read Status Register (70H)

This is one of the two commands that is executable while the state machine is operating. After this command is written, a read of the device will output the contents of the status register, regardless of the address presented to the device.

The device automatically enters this mode after program or erase has completed.

Clear Status Register (50H)

The WSM can only set the Program Status and Erase Status bits in the status register, it can not clear them. Two reasons exist for operating the status register in this fashion. The first is a synchronization. The WSM does not know when the host CPU has read the status register, therefore it would not know when to clear the status bits. Secondly, if the CPU is programming a string of bytes, it may be more efficient to query the status register after programming the string. Thus, if any errors exist while programming the string, the status register will return the accumulated error status.

Program Setup (40H or 10H)

This command simply sets the CUI into a state such that the next write will load the address and data registers. Either 40H or 10H can be used for Program Setup. Both commands are included to accommodate efforts to achieve an industry standard command code set.

Program

The second write after the program setup command, will latch addresses and data. Also, the CUI initiates the WSM to begin execution of the program algorithm. While the WSM finishes the algorithm, the device will output Status Register contents. Note that the WSM cannot be suspended during programming.

Erase Setup (20H)

Prepares the CUI for the Erase Confirm command. No other action is taken. If the next command is not an Erase Confirm command then the CUI will set both the Program Status and Erase Status bits of the Status Register to a "1", place the device into the Read Status Register state, and wait for another command.

Erase Confirm (D0H)

If the previous command was an Erase Setup command, then the CUI will enable the WSM to erase, at the same time closing the address and data latches, and respond only to the Read Status Register and Erase Suspend commands. While the WSM is executing, the device will output Status Register data when OE# is toggled low. Status Register data can only be updated by toggling either OE# or CE# low.

Erase Suspend (B0H)

This command only has meaning while the WSM is executing an Erase operation, and therefore will only be responded to during an erase operation. After this command has been executed, the CUI will initiate the WSM to suspend Erase operations, and then return to responding to only Read Status Register or to the Erase Resume commands. Once the WSM has reached the Suspend state, it will set an output into the CUI which allows the CUI to respond to the Read Array, Read Status Register, and Erase Resume commands. In this mode, the CUI will not respond to any other commands. The WSM will also set the WSM Status bit to a "1". The WSM will con-

tinue to run, idling in the SUSPEND state, regardless of the state of all input control pins, with the exclusion of RP#. RP# low will immediately shut down the WSM and the remainder of the chip.

Erase Resume (D0H)

This command will cause the CUI to clear the Suspend state and set the WSM Status bit to a "0", but only if an Erase Suspend command was previously issued. Erase Resume will not have any effect in all other conditions.

4.4.3 STATUS REGISTER

The 4 Mbit flash family contains a status register which may be read to determine when a program or erase operation is complete, and whether that operation completed successfully. The status register may be read at any time by writing the Read Status command to the CUI. After writing this command, all subsequent Read operations output data from the status register until another command is written to the CUI. A Read Array command must be written to the CUI to return to the Read Array mode.

The status register bits are output on DQ[0:7] whether the device is in the byte-wide (x8) or word-wide (x16) mode for the 28F400BX-L. In the word-wide mode the upper byte, DQ[8:15] is set to 00H during a Read Status command. In the byte-wide mode, DQ[8:14] is tri-stated and DQ15/A-1 retains the low order address function.

It should be noted that the contents of the status register are latched on the falling edge of OE# or CE# whichever occurs last in the read cycle. This prevents possible bus errors which might occur if the contents of the status register change while reading the status register. CE# or OE# must be toggled with each subsequent status read, or the completion of a program or erase operation will not be evident.

The Status Register is the interface between the microprocessor and the Write State Machine (WSM). When the WSM is active, this register will indicate the status of the WSM, and will also hold the bits indicating whether or not the WSM was successful in performing the desired operation. The WSM sets status bits "Three" through "Seven" and clears bits "Six" and "Seven", but cannot clear status bits "Three" through "Five". These bits can only be cleared by the controlling CPU through the use of the Clear Status Register command.

4.4.3.1 Status Register Bit Definition

Table 5. Status Register Definitions

WSMS	ESS	ES	PS	VPPS	R	R	R
7	6	5	4	3	2	1	0

NOTES:

Write State Machine Status bit must first be checked to determine byte/word program or block erase completion, before the Program or Erase Status bits are checked for success.

When Erase Suspend is issued, WSM halts execution and sets both WSMS and ESS bits to "1". ESS bit remains set to "1" until an Erase Resume command is issued.

When this bit is set to "1", WSM has applied the maximum number of erase pulses to the block and is still unable to successfully perform an erase verify.

When this bit is set to "1", WSM has attempted but failed to Program a byte or word.

The Vpp Status bit unlike an A/D converter, does not provide continuous indication of Vpp level. The WSM interrogates the Vpp level only after the byte write or block erase command sequences have been entered and informs the system if Vpp has not been switched on. The Vpp Status bit is not guaranteed to report accurate feedback between VppL and VppH.

SR.7 = WRITE STATE MACHINE STATUS
1 = Ready
0 = Busy

SR.6 = ERASE SUSPEND STATUS
1 = Erase Suspend
0 = Erase in Progress/Completed

SR.5 = ERASE STATUS
1 = Error in Block Erasure
0 = Successful Block Erase

SR.4 = PROGRAM STATUS
1 = Error in Byte/Word Program
0 = Successful Byte/Word Program

SR.3 = Vpp STATUS
1 = Vpp Low Detect; Operation Abort
0 = Vpp OK

SR.2-SR.0 = RESERVED FOR FUTURE ENHANCEMENTS
These bits are reserved for future use and should be masked out when polling the Status Register.

4.4.3.2 Clearing the Status Register

Certain bits in the status register are set by the write state machine, and can only be reset by the system software. These bits can indicate various failure conditions. By allowing the system software to control the resetting of these bits, several operations may be performed (such as cumulatively programming several bytes or erasing multiple blocks in sequence). The status register may then be read to determine if an error occurred during that programming or erasure series. This adds flexibility to the way the device may be programmed or erased. To clear the status register, the Clear Status Register command is written to the CUI. Then, any other command may be issued to the CUI. Note again that before a read cycle can be initiated, a Read command must be written to the CUI to specify whether the read data is to come from the array, status register, or Intelligent Identifier.

4.4.4 PROGRAM MODE

Program is executed by a two-write sequence. The Program Setup command is written to the CUI followed by a second write which specifies the address and data to be programmed. The write state machine will execute a sequence of internally timed events to:

1. program the desired bits of the addressed memory word (byte), and
2. verify that the desired bits are sufficiently programmed.

Programming of the memory results in specific bits within a byte or word being changed to a "0".

If the user attempts to program "1"s, there will be no change of the memory cell content and no error occurs.

Similar to erasure, the status register indicates whether programming is complete. While the program sequence is executing, bit 7 of the status register is a "0". The status register can be polled by toggling either CE# or OE# to determine when the program sequence is complete. Only the Read Status Register command is valid while programming is active.

When programming is complete, the status bits, which indicate whether the program operation was successful, should be checked. If the programming operation was unsuccessful, Bit 4 of the status register is set to a "1" to indicate a Program Failure. If Bit 3 is set then V_{pp} was not within acceptable limits, and the WSM will not execute the programming sequence.

The status register should be cleared before attempting the next operation. Any CUI instruction can follow after programming is completed; however, it must be recognized that reads from the memory, status register, or Intelligent Identifier cannot be accomplished until the CUI is given the appropriate command. A Read Array command must first be given before memory contents can be read.

Figure 12 shows a system software flowchart for device byte programming operation. Figure 13 shows a similar flowchart for device word programming operation (28F400BX-L-only).

4.4.5 ERASE MODE

Erasure of a single block is initiated by writing the Erase Setup and Erase Confirm commands to the CUI, along with the addresses, A[12:17] for the 28F400BX-L or A[12:18] for the 28F004BX-L, identifying the block to be erased. These addresses are latched internally when the Erase Confirm command is issued. Block erasure results in all bits within the block being set to "1".

The WSM will execute a sequence of internally timed events to:

1. program all bits within the block
2. verify that all bits within the block are sufficiently programmed
3. erase all bits within the block and
4. verify that all bits within the block are sufficiently erased

While the erase sequence is executing, Bit 7 of the status register is a "0".

When the status register indicates that erasure is complete, the status bits, which indicate whether the erase operation was successful, should be checked. If the erasure operation was unsuccessful, Bit 5 of the status register is set to a "1" to indicate an Erase Failure. If V_{pp} was not within acceptable limits after the Erase Confirm command is issued, the WSM will not execute an erase sequence; instead, Bit 5 of the status register is set to a "1" to indicate an Erase Failure, and Bit 3 is set to a "1" to identify that V_{pp} supply voltage was not within acceptable limits.

The status register should be cleared before attempting the next operation. Any CUI instruction can follow after erasure is completed; however, it must be recognized that reads from the memory array, status register, or Intelligent Identifier can not be accomplished until the CUI is given the appropriate command. A Read Array command must first be given before memory contents can be read.

Figure 13 shows a system software flowchart for Block Erase operation.

4.4.5.1 Suspending and Resuming Erase

Since an erase operation typically requires 2 seconds to 5 seconds to complete, an Erase Suspend command is provided. This allows erase-sequence interruption in order to read data from another block of the memory. Once the erase sequence is started, writing the Erase Suspend command to the CUI requests that the Write State Machine (WSM) pause the erase sequence at a predetermined point in the erase algorithm. The status register must be read to determine when the erase operation has been suspended.

At this point, a Read Array command can be written to the CUI in order to read data from blocks other than that which is being suspended. The only other valid command at this time is the Erase Resume command or Read Status Register operation.

Figure 14 shows a system software flowchart detailing the operation.

During Erase Suspend mode, the chip can go into a pseudo-standby mode by taking CE# to V_{IH} and the active current is now a maximum of 6 mA. If the chip is enabled while in this mode by taking CE# to V_{IL}, the Erase Resume command can be issued to resume the erase operation.

Upon completion of reads from any block other than the block being erased, the Erase Resume command must be issued. When the Erase Resume command is given, the WSM will continue with the erase sequence and complete erasing the block. As with the end of erase, the status register must be read, cleared, and the next instruction issued in order to continue.

The status register should be cleared before attempting the next operation. Any CPU instruction can follow after erase is completed; however, it must be recognized that reads from the memory array, status register, or intelligent identifier can not be accomplished until the CUI is given the appropriate command. A Read Array command must be given before memory contents can be read.

Figure 13 shows a system software flowchart for Block Erase operation.

4.4.3.1 Suspending and Resuming Erase

Since an erase operation typically requires 2 seconds to complete, an Erase Suspend command is provided. This allows the user to suspend the erase sequence in order to read data from another block of the memory. Once the erase sequence is resumed by writing the Erase Suspend command in the CUI, the device will resume the erase sequence at a predetermined point in the erase algorithm. The status register must be read to determine when the erase operation has been suspended.

At this point, a Read Array command can be written to the CUI in order to read data from block other than that which is being suspended. The only other valid command at this time is the Erase Resume command or Read Status Register operation.

Figure 14 shows a system software flowchart detailing the operation.

During Erase Suspend mode, the chip can go into a pseudo-idle mode by taking CE₁ to V_{DD} and the active current is now a maximum of 10 mA. If the chip is enabled while in this mode by taking CE₁ to V_{DD}, the Erase Resume command can be issued to resume the erase operation.

4.4.6 EXTENDED CYCLING

Intel has designed extended cycling capability into its ETOX III flash memory technology. The 4-Mbit low voltage flash memory family is designed for 10,000 program/erase cycles on each of the seven blocks. The combination of low electric fields, clean oxide processing and minimized oxide area per memory cell subjected to the tunneling electric field, results in very high cycling capability.

The status register should be cleared before attempting the next operation. Any CPU instruction can follow after programming is completed; however, it must be recognized that reads from the memory array, status register, or intelligent identifier cannot be accomplished until the CUI is given the appropriate command. A Read Array command must be given before memory contents can be read.

Figure 12 shows a system software flowchart for device word programming operation. Figure 13 shows a similar flowchart for device word programming operation (28F400BX-TL only).

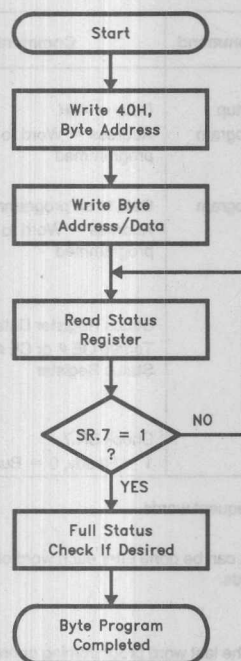
4.4.8 ERASE MODE

Erase of a single block is initiated by writing the Erase Setup and Erase Confirm commands to the CUI, along with the addresses A[12:7] for the 28F400BX-TL (A[12:5]) for the 28F004BX-TL. These addresses are the ones to be erased. These addresses are latched internally when the Erase Confirm command is issued. Block erase results in all bits within the block being set to "1".

The WSM will execute a sequence of internally timed events to:

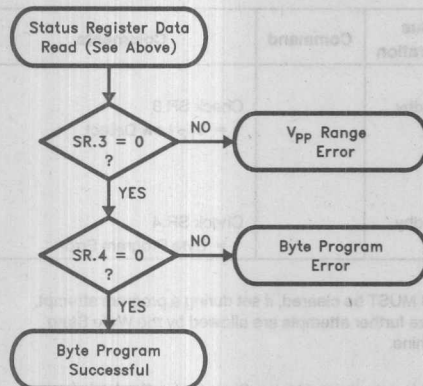
1. program all bits within the block.
2. verify that all bits within the block are sufficiently programmed.
3. erase all bits within the block and
4. verify that all bits within the block are sufficiently erased.

While the erase sequence is executing, Bit 7 of the status register is a "0".



290450-9

Full Status Check Procedure



290450-10

Bus Operation	Command	Comments
Write	Setup Program	Data = 40H Address = Byte to be programmed
Write	Program	Data to be programmed Address = Byte to be programmed
Read		Status Register Data. Toggle OE# or CE# to update Status Register
Standby		Check SR.7 1 = Ready, 0 = Busy

Repeat for subsequent bytes.

Full status check can be done after each byte or after a sequence of bytes.

Write FFH after the last byte programming operation to reset the device to Read Array Mode.

Bus Operation	Command	Comments
Standby		Check SR.3 1 = Vpp Low Detect
Standby		Check SR.4 1 = Byte Program Error

SR.3 MUST be cleared, if set during a program attempt, before further attempts are allowed by the Write State Machine.

SR.4 is only cleared by the Clear Status Register Command, in cases where multiple bytes are programmed before full status is checked.

If error is detected, clear the Status Register before attempting retry or other error recovery.

Figure 12. Automated Byte Programming Flowchart

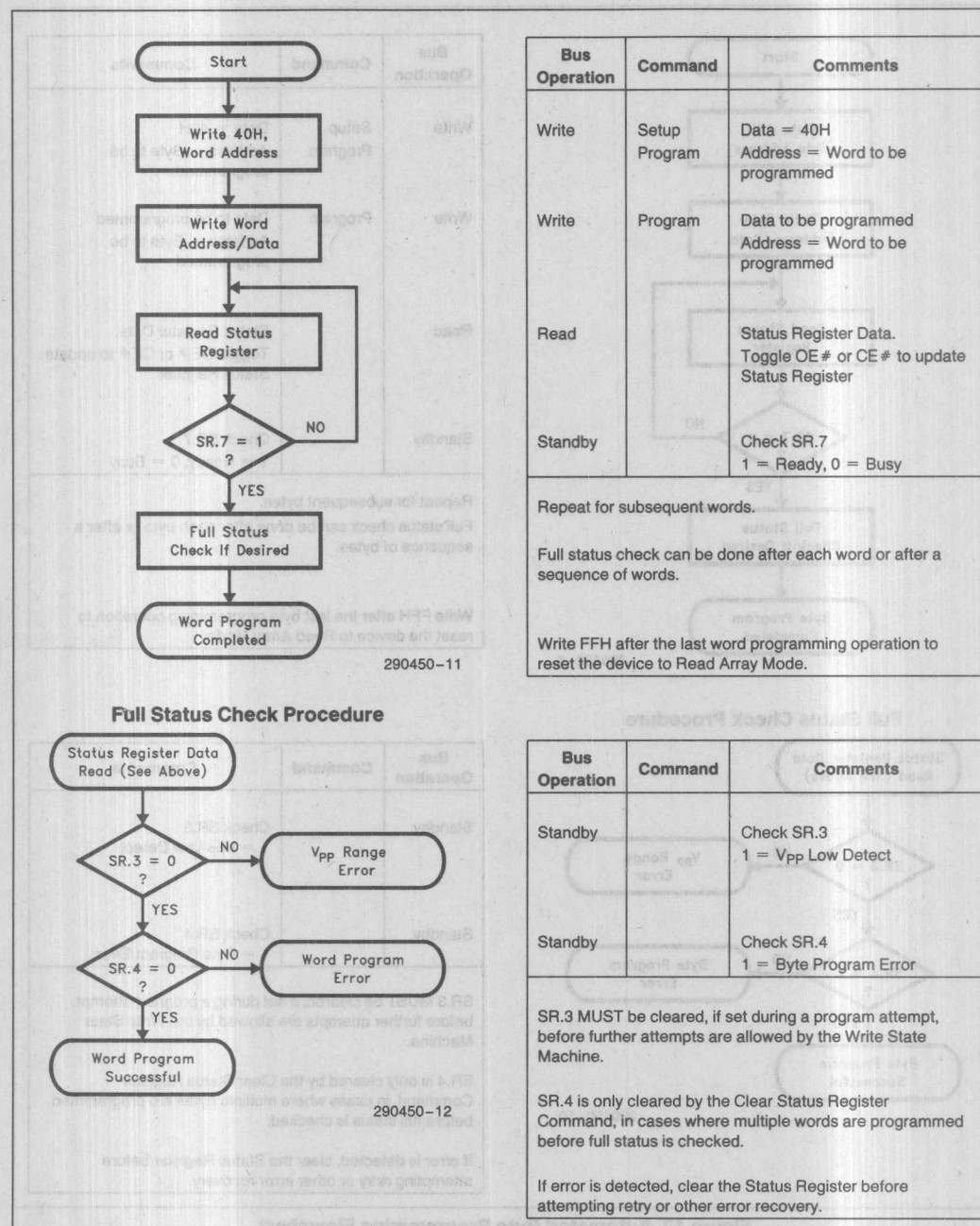
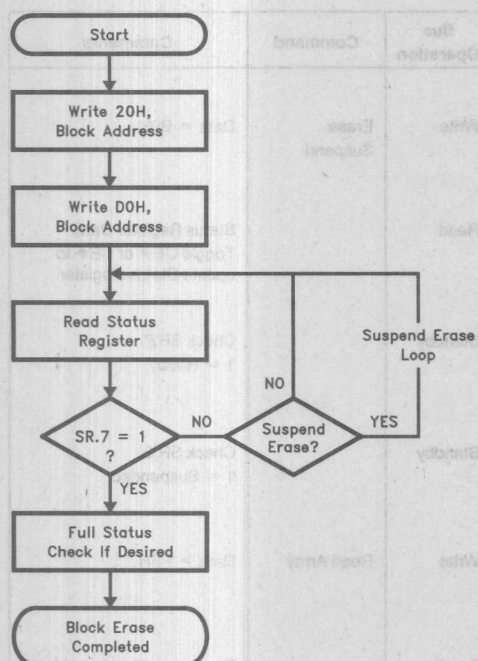
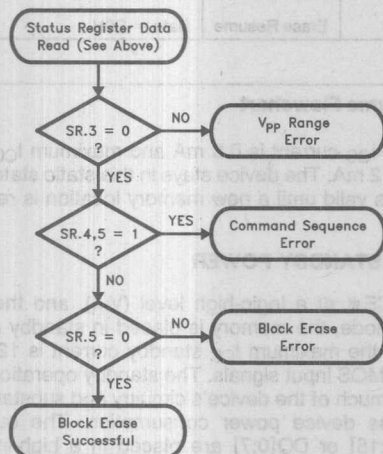


Figure 13. Automated Word Programming Flowchart



Full Status Check Procedure



Bus Operation	Command	Comments
Write	Setup Erase	Data = 20H Address = Within block to be erased
Write	Erase	Data = D0H Address = Within block to be erased
Read		Status Register Data. Toggle OE# or CE# to update Status Register
Standby		Check SR.7 1 = Ready, 0 = Busy

Repeat for subsequent blocks.

Full status check can be done after each block or after a sequence of blocks.

Write FFH after the last block erase operation to reset the device to Read Array Mode.

Bus Operation	Command	Comments
Standby		Check SR.3 1 = Vpp Low Detect
Standby		Check SR.4,5 Both 1 = Command Sequence Error
Standby		Check SR.5 1 = Block Erase Error

SR.3 MUST be cleared, if set during an erase attempt, before further attempts are allowed by the Write State Machine.

SR.5 is only cleared by the Clear Status Register Command, in cases where multiple blocks are erased before full status is checked.

If error is detected, clear the Status Register before attempting retry or other error recovery.

Figure 14. Automated Block Erase Flowchart

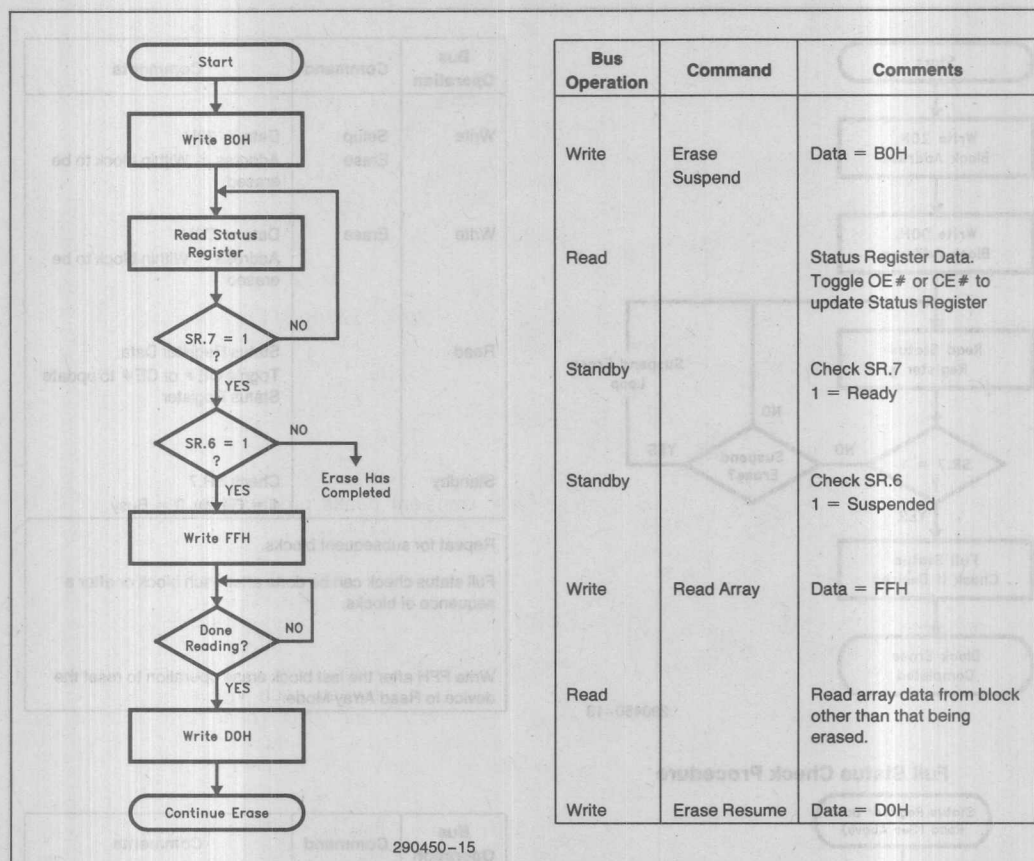


Figure 15. Erase Suspend/Resume Flowchart

4.5 Power Consumption

4.5.1 ACTIVE POWER

With **CE#** at a logic-low level and **RP#** at a logic-high level, the device is placed in the active mode. The device I_{CC} current is a maximum of 22 mA at 5 MHz.

4.5.2 AUTOMATIC POWER SAVINGS

Automatic Power Savings (APS) is a low power feature during active mode of operation. The 4-Mbit family of products incorporate Power Reduction Control (PRC) circuitry which basically allows the device to put itself into a low current state when it is not being accessed. After data is read from the memory array, PRC logic controls the device's power consumption by entering the APS mode where

typical I_{CC} current is 0.8 mA and maximum I_{CC} current is 2 mA. The device stays in this static state with outputs valid until a new memory location is read.

4.5.3 STANDBY POWER

With **CE#** at a logic-high level (V_{IH}), and the **CUI** read mode, the memory is placed in standby mode where the maximum I_{CC} standby current is 120 μ A with CMOS input signals. The standby operation disables much of the device's circuitry and substantially reduces device power consumption. The outputs (DQ[0:15] or DQ[0:7]) are placed in a high-impedance state independent of the status of the **OE#** signal. When the 4-Mbit flash family is deselected during erase or program functions, the devices will continue to perform the erase or program function and consume program or erase active power until program or erase is completed.

4.5.4 RESET/DEEP POWER-DOWN

The 4-Mbit flash family supports a typical I_{CC} of 0.2 μA in deep power-down mode. One of the target markets for these devices is in portable equipment where the power consumption of the machine is of prime importance. The 4-Mbit flash family has a RP# pin which places the device in the deep power-down mode. When RP# is at a logic-low (GND $\pm 0.2V$), all circuits are turned off and the device typically draws 0.2 μA of V_{CC} current.

During read modes, the RP# pin going low deselects the memory and places the output drivers in a high impedance state. Recovery from the deep power-down state, requires a minimum of 700 ns to access valid data (t_{PHQV}).

During erase or program modes, RP# low will abort either erase or program operation. The contents of the memory are no longer valid as the data has been corrupted by the RP# function. As in the read mode above, all internal circuitry is turned off to achieve the 0.2 μA current level.

RP# transitions to V_{IL} or turning power off to the device will clear the status register.

This use of RP# during system reset is important with automated write/erase devices. When the system comes out of reset it expects to read from the flash memory. Automated flash memories provide status information when accessed during write/erase modes. If a CPU reset occurs with no flash memory reset, proper CPU initialization would not occur because the flash memory would be providing the status information instead of array data. Intel's Flash Memories allow proper CPU initialization following a system reset through the use of the RP# input. In this application RP# is controlled by the same RESET# signal that resets the system CPU.

4.6 Power-up Operation

The 4-Mbit flash memory family is designed to offer protection against accidental block erasure or programming during power transitions. Upon power-up the 4-Mbit flash memory family is indifferent as to which power supply, V_{PP} or V_{CC} , powers-up first. Power supply sequencing is not required.

The 4-Mbit flash memory family ensures the CUI is reset to the read mode on power-up.

In addition, on power-up the user must either drop CE# low or present a new address to ensure valid data at the outputs.

A system designer must guard against spurious writes for V_{CC} voltages above V_{LKO} when V_{PP} is

active. Since both WE# and CE# must be low for a command write, driving either signal to V_{IH} will inhibit writes to the device. The CUI architecture provides an added level of protection since alteration of memory contents can only occur after successful completion of the two-step command sequences. Finally the device is disabled until RP# is brought to V_{IH} , regardless of the state of its control inputs. This feature provides yet another level of memory protection.

4.7 Power Supply Decoupling

Flash memory's power switching characteristics require careful device decoupling methods. System designers are interested in 3 supply current issues:

- Standby current levels (I_{CCS})
- Active current levels (I_{CCR})
- Transient peaks produced by falling and rising edges of CE#.

Transient current magnitudes depend on the device outputs' capacitive and inductive loading. Two-line control and proper decoupling capacitor selection will suppress these transient voltage peaks. Each flash device should have a 0.1 μF ceramic capacitor connected between each V_{CC} and GND, and between its V_{PP} and GND. These high frequency, low-inductance capacitors should be placed as close as possible to the package leads.

4.7.1 V_{PP} TRACE ON PRINTED CIRCUIT BOARDS

Writing to flash memories while they reside in the target system, requires special consideration of the V_{PP} power supply trace by the printed circuit board designer. The V_{PP} pin supplies the flash memory cell's current for programming and erasing. One should use similar trace widths and layout considerations given to the V_{CC} power supply trace. Adequate V_{PP} supply traces and decoupling will decrease spikes and overshoots.

4.7.2 V_{CC} , V_{PP} AND RP# TRANSITIONS

The CUI latches commands as issued by system software and is not altered by V_{PP} or CE# transitions or WSM actions. Its state upon power-up, after exit from deep power-down mode or after V_{CC} transitions below V_{LKO} (Lockout voltage), is Read Array mode.

After any word/byte write or block erase operation is complete and even after V_{PP} transitions down to V_{PPL} , the CUI must be reset to Read Array mode via the Read Array command when accesses to the flash memory are desired.

5.0 OPERATING SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS

Operating Temperature	
During Read0°C to +70°C(1)
During Block Erase/Byte Write0°C to +70°C
Temperature Under Bias-10°C to +80°C
Storage Temperature-65°C to +125°C
Voltage on any Pin	
(except V _{CC} , V _{PP} , A ₉ and RP#)	
with Respect to GND-2.0V to +7.0V(2)
Voltage on Pin RP# or Pin A ₉	
with Respect to GND-2.0V to +13.5V(2,3)
V _{CC} Program Voltage with	
Respect to GND	
during Block Erase and	
Word/Byte Write-2.0V to +14.0V(2,3)
V _{CC} Supply Voltage	
with Respect to GND-2.0V to +7.0V(2)
Output Short Circuit Current100 mA(4)

NOTES:

1. Operating temperature is for commercial product defined by this specification.
2. Minimum DC voltage is -0.5V on input/output pins. During transitions, this level may undershoot to -2.0V for periods <20 ns. Maximum DC voltage on input/output pins is V_{CC} + 0.5V which during transitions may overshoot to V_{CC} + 2.0V for periods < 20 ns.
3. Maximum DC voltage on V_{PP} may overshoot to +14.0V for periods <20 ns. Maximum DC voltage on RP# or A₉ may overshoot to 13.5V for periods < 20 ns.
4. Output shorted for no more than one second. No more than one output shorted at a time.
5. AC specifications are valid at both voltage ranges. See DC Characteristics tables for voltage range-specific specifications.

OPERATING CONDITIONS

Symbol	Parameter	Notes	Min	Max	Unit
T _A	Operating Temperature		0	70	°C
V _{CC}	V _{CC} Supply Voltage	5	3.00	3.60	V
V _{CC}	V _{CC} Supply Voltage	5	4.50	5.50	V

DC CHARACTERISTICS V_{CC} = 3.3V ± 0.3V

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Conditions
I _{LI}	Input Load Current	1			±1.0	μA	V _{CC} = V _{CC} Max V _{IN} = V _{CC} or GND
I _{LO}	Output Leakage Current	1			±10	μA	V _{CC} = V _{CC} Max V _{OUT} = V _{CC} or GND
I _{CCS}	V _{CC} Standby Current	1, 3		45	120	μA	V _{CC} = V _{CC} Max CE# = RP# = V _{IH}
				45	120	μA	V _{CC} = V _{CC} Max CE# = RP# = V _{CC} ± 0.2V 28F400BX: BYTE# = V _{CC} ± 0.2V or GND

NOTICE: This data sheet contains preliminary information on new products in production. The specifications are subject to change without notice. Verify with your local Intel Sales office that you have the latest data sheet before finalizing a design.

***WARNING:** Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.

D.C. CHARACTERISTICS $V_{CC} = 3.3V \pm 0.3V$ (Continued)

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Conditions
I_{CCD}	V_{CC} Deep Power-Down Current	1		0.20	1.2	μA	RP# = GND $\pm 0.2V$
I_{CCR}	V_{CC} Read Current for 28F400BX-L Word-Wide and Byte-Wide Mode and 28F004BX-L Byte-Wide Mode	1 5, 6		15	25	mA	$V_{CC} = V_{CC} \text{ Max}$, CE# = GND f = 5 MHz, $I_{OUT} = 0 \text{ mA}$ CMOS Inputs
				15	25	mA	$V_{CC} = V_{CC} \text{ Max}$, CE# = V_{IL} f = 5 MHz, $I_{OUT} = 0 \text{ mA}$ TTL Inputs
I_{CCW}	V_{CC} Word Write Current	1			30	mA	Word Write in Progress
I_{CCW}	V_{CC} Byte Write Current	1			30	mA	Byte Write in Progress
I_{CCE}	V_{CC} Block Erase Current	1			20	mA	Block Erase in Progress
I_{CCES}	V_{CC} Erase Suspend Current	1, 2		3	6	mA	Block Erase Suspended, CE# = V_{IH}
I_{PPS}	V_{PP} Standby Current	1			± 15	μA	$V_{PP} \leq V_{CC}$
I_{PPD}	V_{PP} Deep Power-Down Current	1			5.0	μA	RP# = GND $\pm 0.2V$
I_{PPR}	V_{PP} Read Current	1			200	μA	$V_{PP} > V_{CC}$
I_{PPW}	V_{PP} Word Write Current	1			40	mA	$V_{PP} = V_{PPH}$ Word Write in Progress
I_{PPW}	V_{PP} Byte Write Current	1			30	mA	$V_{PP} = V_{PPH}$ Byte Write in Progress
I_{PPE}	V_{PP} Block Erase Current	1			30	mA	$V_{PP} = V_{PPH}$ Block Erase in Progress
I_{PPES}	V_{PP} Erase Suspend Current	1			200	μA	$V_{PP} = V_{PPH}$ Block Erase Suspended
$I_{RP\#}$	RP# Boot Block Unlock Current	1, 4			500	μA	RP# = V_{HH}
I_{ID}	A ₉ Intelligent Identifier Current	1, 4			500	μA	A ₉ = V_{ID}
V_{ID}	A ₉ Intelligent Identifier Voltage		11.4	12.0	13.0	V	
V_{IL}	Input Low Voltage		-0.5		0.6	V	
V_{IH}	Input High Voltage		2.0		$V_{CC} + 0.5$	V	
V_{OL}	Output Low Voltage				0.4	V	$V_{CC} = V_{CC} \text{ Min}$ $I_{OL} = 2 \text{ mA}$
V_{OH}	Output High Voltage		2.4			V	$V_{CC} = V_{CC} \text{ Min}$ $I_{OH} = -2 \text{ mA}$
V_{PPL}	V_{PP} during Normal Operations	3	0.0		4.1	V	
V_{PPH}	V_{PP} during Erase/Write Operations		11.4	12.0	12.6	V	
V_{LKO}	V_{CC} Erase/Write Lock Voltage		2.0			V	
V_{HH}	RP# Unlock Voltage		11.4	12.0	13.0	V	Boot Block Write/Erase

CAPACITANCE⁽⁴⁾ $T_A = 25^\circ\text{C}$, $f = 1\text{ MHz}$

Symbol	Parameter	Typ	Max	Unit	Condition
C_{IN}	Input Capacitance	6	8	pF	$V_{IN} = 0\text{V}$
C_{OUT}	Output Capacitance	10	12	pF	$V_{OUT} = 0\text{V}$

NOTES:

1. All currents are in RMS unless otherwise noted. Typical values at $V_{CC} = 3.3\text{V}$, $V_{PP} = 12.0\text{V}$, $T = 25^\circ\text{C}$. These currents are valid for all product versions (packages and speeds).
2. I_{CCES} is specified with the device deselected. If the device is read while in Erase Suspend Mode, current draw is the sum of I_{CCES} and I_{CCR} .
3. Block Erase and Word/Byte Writes are inhibited when $V_{PP} = V_{PPL}$ and not guaranteed in the range between V_{PPH} and V_{PPL} .
4. Sampled, not 100% tested.
5. Automatic Power Savings (APS) reduces I_{CCR} to less than 2 mA in static operation.
6. CMOS Inputs are either $V_{CC} \pm 0.2\text{V}$ or $\text{GND} \pm 0.2\text{V}$. TTL Inputs are either V_{IL} or V_{IH} .

I_{CCES}	VCC Erase Suspend Current	1.2	3	8	mA Block Erase Suspend Current
I_{CCR}	VCC Read Current	1	1.5	1.5	µA VPP = VCC
I_{PPL}	VPP Standby Current	1	1	1.5	µA VPP = VCC
I_{PDS}	VPP Power-Down Current	1	1	50	µA VPP = VCC ± 0.2V
I_{PWS}	VPP Write Current	1	1	500	µA VPP = VCC
I_{PWW}	VPP Word Write Current	1	1	40	mA VPP = VCC
I_{PBYW}	VPP Byte Write Current	1	1	30	mA VPP = VCC
I_{PEES}	VPP Erase Suspend Current	1	1	30	mA VPP = VCC
I_{PES}	VPP Erase Suspend Current	1	1	200	µA VPP = VCC
I_{PUS}	VPP Block Unlock Current	1.4	1.4	500	µA VPP = VCC
I_{AI}	AI Intelligent Identifier Current	1.4	1.4	500	µA VCC = VCC
V_{IO}	AI Intelligent Identifier Voltage	1.7 ± 0.5	1.7 ± 0.5	1.7 ± 0.5	V
V_{IL}	Input Low Voltage	-0.5	0.8	0.8	V
V_{IH}	Input High Voltage	2.0	2.0	2.0	VCC + 0.2 V
V_{OL}	Output Low Voltage	0.4	0.4	0.4	VCC = VCC
V_{OH}	Output High Voltage	2.4	2.4	2.4	VCC = VCC
V_{T}	VCC Standby Internal Operations	3	0.0	0.1	V
V_{E}	VCC Erase/Write Operations	1.4 ± 0.5	1.4 ± 0.5	1.4 ± 0.5	V
V_{LW}	VCC Standby/Write Lock Voltage	2.0	2.0	2.0	V
V_{P}	VPP Block Write Voltage	1.7 ± 0.5	1.7 ± 0.5	1.7 ± 0.5	V

D.C. CHARACTERISTICS(4) $V_{CC} = 5.0V \pm 10\%$

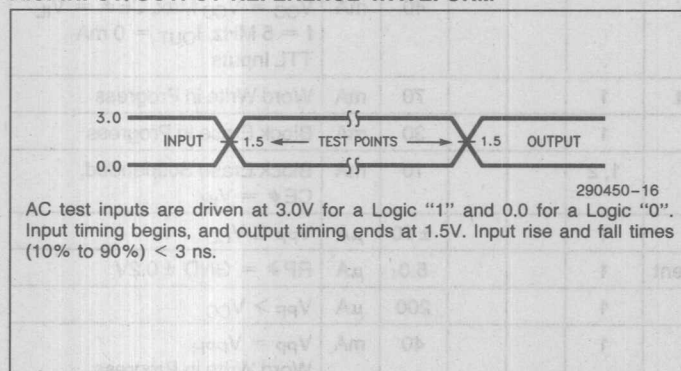
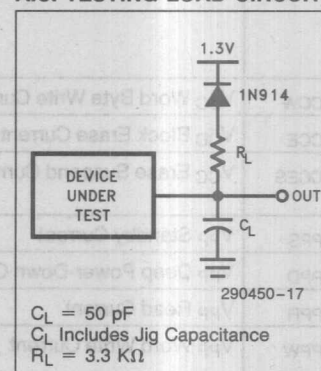
Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Conditions
I_{LI}	Input Load Current	1			± 1.0	μA	$V_{CC} = V_{CC} \text{ Max}$ $V_{IN} = V_{CC} \text{ or GND}$
I_{LO}	Output Leakage Current	1			± 10	μA	$V_{CC} = V_{CC} \text{ Max}$ $V_{OUT} = V_{CC} \text{ or GND}$
I_{CCS}	V_{CC} Standby Current	1			1.5	mA	$V_{CC} = V_{CC} \text{ Max}$ $CE\# = RP\# = V_{IH}$
					100	μA	$V_{CC} = V_{CC} \text{ Max}$ $CE\# = RP\# = V_{CC} \pm 0.2V$
I_{CCD}	V_{CC} Deep Power-Down Current	1			1.2	μA	$RP\# = GND \pm 0.2V$ $I_{OUT} = 0 \text{ mA}$
I_{CCR}	V_{CC} Read Current for 28F400BX-L Word-Wide Mode	1			45	mA	$V_{CC} = V_{CC} \text{ Max}$, $CE\# = GND$ $f = 5 \text{ MHz}$, $I_{OUT} = 0 \text{ mA}$ CMOS Inputs
					45	mA	$V_{CC} = V_{CC} \text{ Max}$, $CE\# = V_{IL}$ $f = 5 \text{ MHz}$, $I_{OUT} = 0 \text{ mA}$ TTL Inputs
I_{CCR}	V_{CC} Read Current for 28F400BX-L Byte-Wide Mode and 28F004BX-L	1			40	mA	$V_{CC} = V_{CC} \text{ Max}$, $CE\# = GND$ $f = 5 \text{ MHz}$, $I_{OUT} = 0 \text{ mA}$ CMOS Inputs
					40	mA	$V_{CC} = V_{CC} \text{ Max}$, $CE\# = V_{IL}$ $f = 5 \text{ MHz}$, $I_{OUT} = 0 \text{ mA}$ TTL Inputs
I_{CCW}	V_{CC} Word Byte Write Current	1			70	mA	Word Write in Progress
I_{CCE}	V_{CC} Block Erase Current	1			30	mA	Block Erase in Progress
I_{CCES}	V_{CC} Erase Suspend Current	1, 2			10	mA	Block Erase Suspended, $CE\# = V_{IH}$
I_{PPS}	V_{PP} Standby Current	1			± 10	μA	$V_{PP} \leq V_{CC}$
I_{PPD}	V_{PP} Deep Power-Down Current	1			5.0	μA	$RP\# = GND \pm 0.2V$
I_{PPR}	V_{PP} Read Current	1			200	μA	$V_{PP} > V_{CC}$
I_{PPW}	V_{PP} Word Write Current	1			40	mA	$V_{PP} = V_{PPH}$ Word Write in Progress
I_{PPW}	V_{PP} Byte Write Current	1			30	mA	$V_{PP} = V_{PPH}$ Byte Write in Progress
I_{PPE}	V_{PP} Block Erase Current	1			30	mA	$V_{PP} = V_{PPH}$ Block Erase in Progress
I_{PPES}	V_{PP} Erase Suspend Current	1			200	μA	$V_{PP} = V_{PPH}$ Block Erase Suspended
$I_{RP\#}$	$RP\#$ Boot Block Unlock Current	1, 4			500	μA	$RP\# = V_{HH}$
I_{ID}	A_9 Intelligent Identifier Current	1, 4				μA	$A_9 = V_{ID}$
V_{ID}	A_9 Intelligent Identifier Voltage		11.4	12.0	13.0	V	

D.C. CHARACTERISTICS(4) $V_{CC} = 5.0V \pm 10\%$ (Continued)

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Condition
V_{IL}	Input Low Voltage		-0.5		0.8	V	
V_{IH}	Input High Voltage		2.0		$V_{CC} + 0.5$	V	
V_{OL}	Output Low Voltage				0.45	V	$V_{CC} = V_{CC} \text{ Min}$ $I_{OL} = 5.8 \text{ mA}$
V_{OH}	Output High Voltage		2.4			V	$V_{CC} = V_{CC} \text{ Min}$ $I_{OH} = -2.5 \text{ mA}$
V_{PPL}	V_{PP} during Normal Operations	3	0.0		6.5	V	
V_{PPH}	V_{PP} during Erase/Write Operations		11.4	12.0	12.6	V	
V_{LKO}	V_{CC} Erase/Write Lock Voltage		2.2			V	
V_{HH}	RP# Unlock Voltage		11.4	12.0	13.0	V	Boot Block Write/Erase

NOTES:

1. All currents are in RMS unless otherwise noted. Typical values at $V_{CC} = 5.0V$, $V_{PP} = 12.0V$, $T = 25^\circ C$. These currents are valid for all product versions (packages and speeds).
2. I_{CCES} is specified with the device deselected. If the device is read while in Erase Suspend Mode, current draw is the sum of I_{CCES} and I_{CCR} .
3. Block Erases and Word/Byte Writes are inhibited when $V_{PP} = V_{PPL}$ and not guaranteed in the range between V_{PPH} and V_{PPL} .
4. All parameters are sampled, not 100% tested.

A.C. INPUT/OUTPUT REFERENCE WAVEFORM**A.C. TESTING LOAD CIRCUIT**

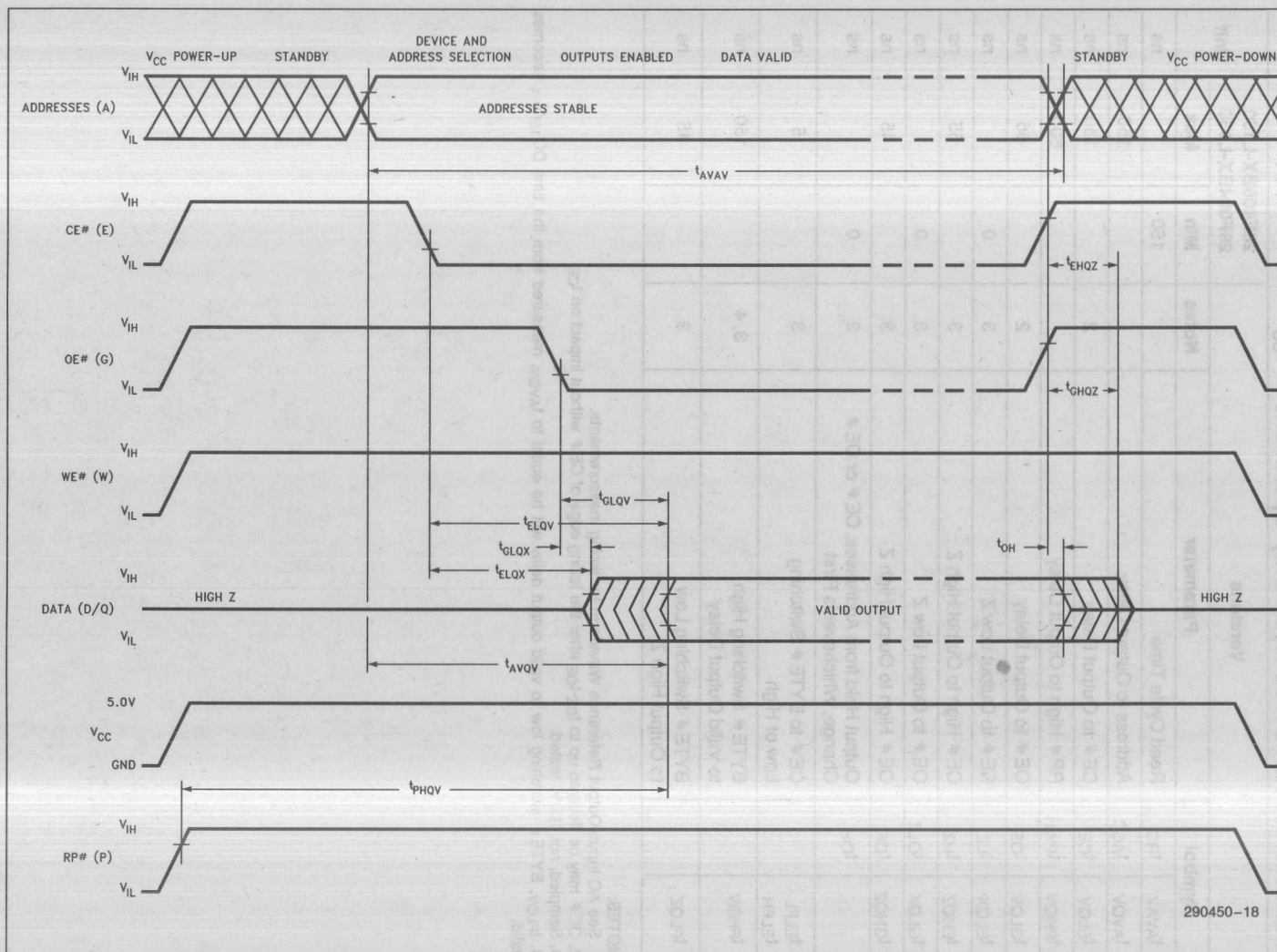
A.C. CHARACTERISTICS-Read-Only Operations⁽¹⁾ $V_{CC} = 3.3V \pm 0.3V, 5.0V \pm 10\%$

Versions			28F400BX-L150 28F004BX-L150		Unit
Symbol	Parameter	Notes	Min	Max	
t_{AVAV} t_{RC}	Read Cycle Time		150		ns
t_{AVQV} t_{ACC}	Address to Output Delay			150	ns
t_{ELQV} t_{CE}	CE # to Output Delay	2		150	ns
t_{PHQV} t_{PWH}	RP # High to Output Delay			600	ns
t_{GLQV} t_{OE}	OE # to Output Delay	2		65	ns
t_{ELQX} t_{LZ}	CE # to Output Low Z	3	0		ns
t_{EHOZ} t_{HZ}	CE # High to Output High Z	3		55	ns
t_{GLQX} t_{OLZ}	OE # to Output Low Z	3	0		ns
t_{GHQZ} t_{DF}	OE # High to Output High Z	3		45	ns
	t_{OH} Output Hold from Addresses, CE # or OE # Change, Whichever is First	3	0		ns
t_{ELFL} t_{ELFH}	CE # to BYTE # Switching Low or High	3		5	ns
t_{FHQV}	BYTE # Switching High to Valid Output Delay	3, 4		150	ns
t_{FLQZ}	BYTE # Switching Low to Output High Z	3		45	ns

NOTES:

1. See AC Input/Output Reference Waveform for timing measurements.
2. OE # may be delayed up to $t_{CE-t_{OE}}$ after the falling edge of CE # without impact on t_{CE} .
3. Sampled, not 100% tested.
4. t_{FLQV} , BYTE # switching low to valid output delay will be equal to t_{AVQV} , measured from the time DQ₁₅/A₁ becomes valid.

Figure 16. A.C. Waveforms for Read Operations



290450-18

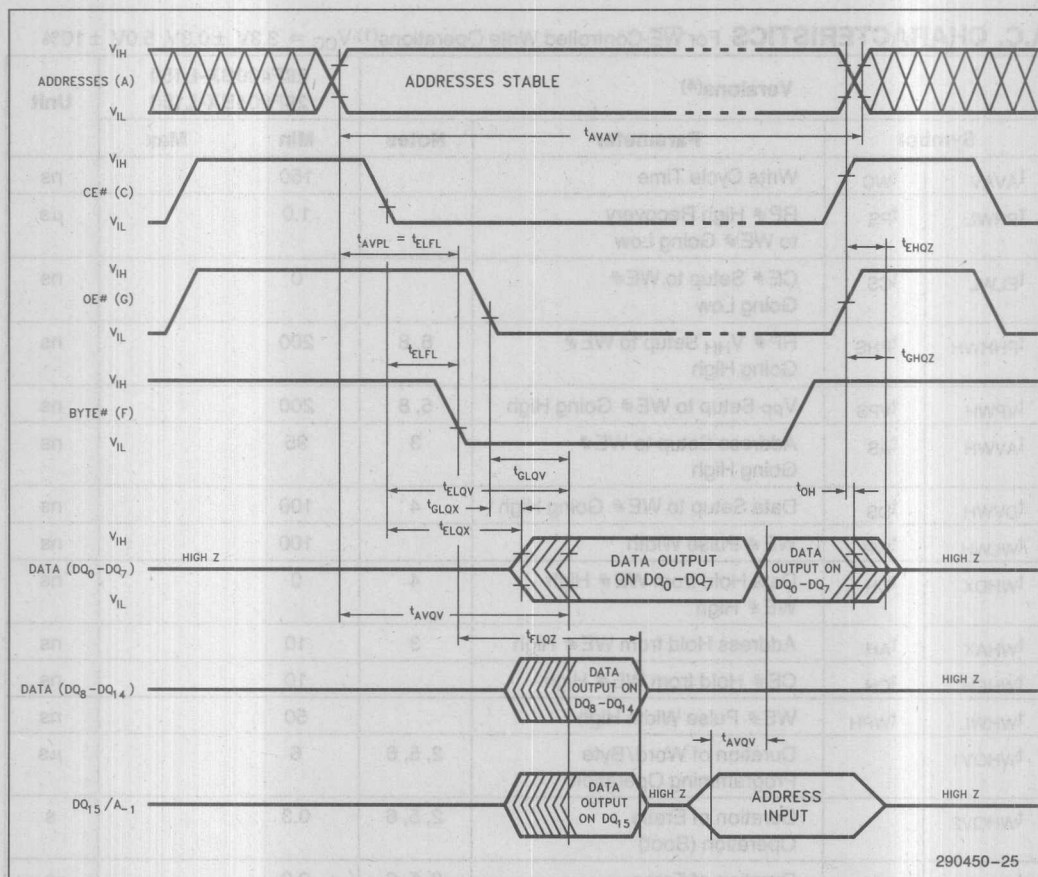


Figure 17. BYTE# Timing Diagram for Both Read and Write Operations for 28F400BX-L

4

1	Whole	Duration of Error	2.5.8	0.8
2	Operation (Main)	Duration of Error	2.5.8	0.8
3	Low	Vpp Hold from Valid SPD	2.5.8	0
4	Low	RP# Vpp Hold from Valid SPD	2.5.8	0
5	Low	Boot/Blank Retack Delay	2.5.8	1.8

NOTES:
 1. Read timing relationships during write and erase operations are the same as during read-only operations. Refer to AD Characterizing Read Mode.
 2. The on-chip ROM controller automatically programs/erases operations program/erase algorithms are not controlled internally, which makes verify and masking operations.
 3. Refer to command definition table for valid A/E.
 4. Refer to command definition table for valid D/E.
 5. Program/erase operations are measured to valid SPD data (successful operation, SR 1-1).
 6. For Boot Blank Program/erase, RWD# should be held at Vpp until operation completes successfully.
 7. Time delay is required for successful relocking of the Boot Block.
 8. Sampled but not 100% tested.

A.C. CHARACTERISTICS For \overline{WE} -Controlled Write Operations⁽¹⁾ $V_{CC} = 3.3V \pm 0.3V, 5.0V \pm 10\%$

Versions ⁽⁴⁾			28F400BX-L150 28F004BX-L150		Unit
Symbol	Parameter	Notes	Min	Max	
t_{AVAV}	t_{WC}	Write Cycle Time	150		ns
t_{PHWL}	t_{PS}	RP# High Recovery to WE# Going Low	1.0		μs
t_{ELWL}	t_{CS}	CE# Setup to WE# Going Low	0		ns
t_{PHHWH}	t_{PHS}	RP# V_{HH} Setup to WE# Going High	6, 8	200	ns
t_{VPWH}	t_{VPS}	V_{PP} Setup to WE# Going High	5, 8	200	ns
t_{AVWH}	t_{AS}	Address Setup to WE# Going High	3	95	ns
t_{DVWH}	t_{DS}	Data Setup to WE# Going High	4	100	ns
t_{WLWH}	t_{WP}	WE# Pulse Width		100	ns
t_{WHDX}	t_{DH}	Data Hold from WE# High WE# High	4	0	ns
t_{WHAX}	t_{AH}	Address Hold from WE# High	3	10	ns
t_{WHEH}	t_{CH}	CE# Hold from WE# High		10	ns
t_{WHWL}	t_{WPH}	WE# Pulse Width High		50	ns
t_{WHQV1}		Duration of Word/Byte Programming Operation	2, 5, 6	6	μs
t_{WHQV2}		Duration of Erase Operation (Boot)	2, 5, 6	0.3	s
t_{WHQV3}		Duration of Erase Operation (Parameter)	2, 5, 6	0.3	s
t_{WHQV4}		Duration of Erase Operation (Main)	2, 5, 6	0.6	s
t_{QWL}	t_{VPH}	V_{PP} Hold from Valid SRD	5, 8	0	ns
t_{QVPH}	t_{PHH}	RP# V_{HH} Hold from Valid SRD	6, 8	0	ns
t_{PHBR}		Boot-Block Relock Delay	7, 8	200	ns

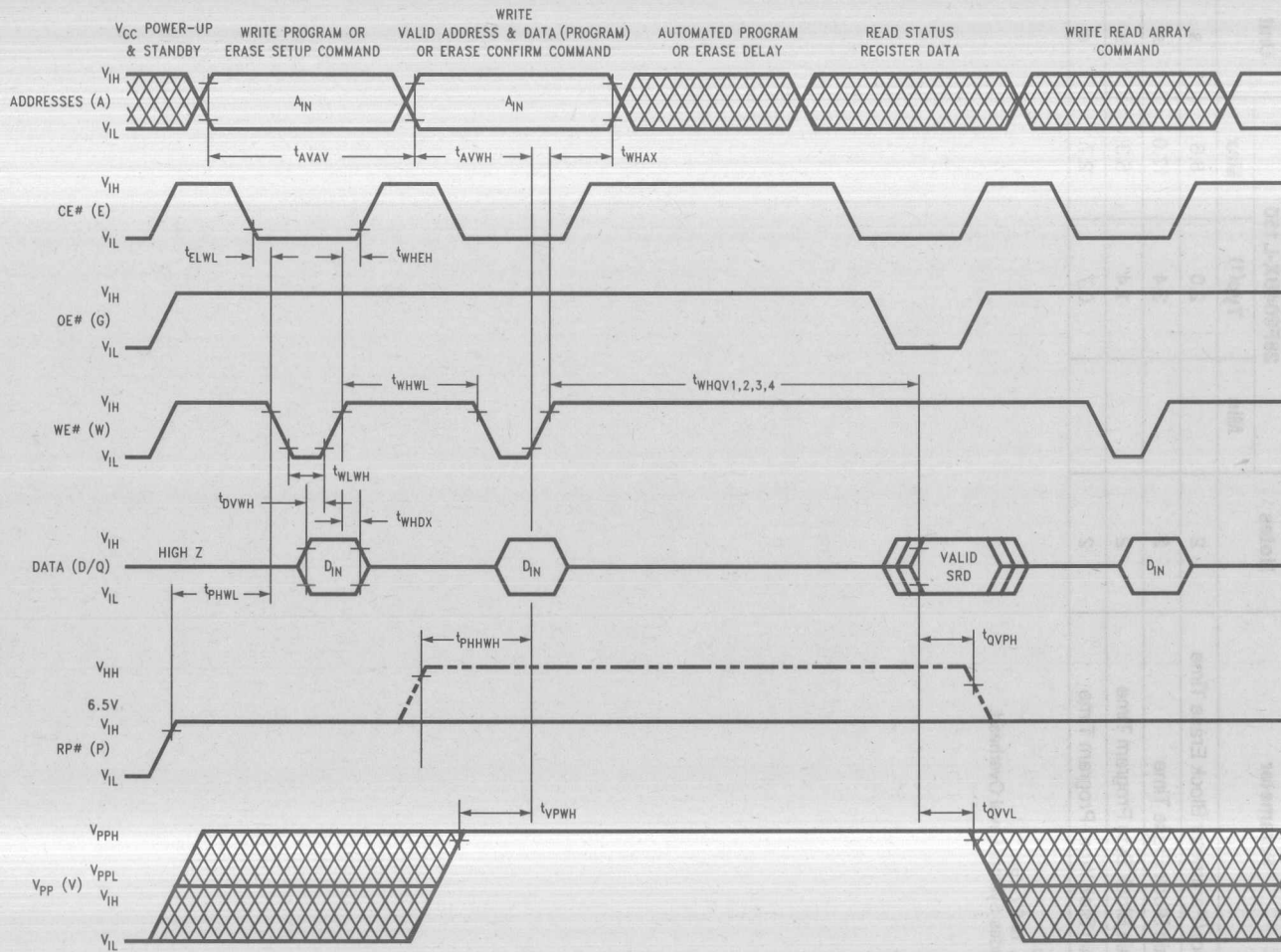
NOTES:

1. Read timing characteristics during write and erase operations are the same as during read-only operations. Refer to AC Characteristics during Read Mode.
2. The on-chip WSM completely automates program/erase operations; program/erase algorithms are now controlled internally which includes verify and margining operations.
3. Refer to command definition table for valid A_{IN} .
4. Refer to command definition table for valid D_{IN} .
5. Program/Erase durations are measured to valid SRD data (successful operation, SR.7 = 1).
6. For Boot Block Program/Erase, PWD# should be held at V_{HH} until operation completes successfully.
7. Time t_{PHBR} is required for successful relocking of the Boot Block.
8. Sampled but not 100% tested.

Parameter	Notes	28F400BX-L150 28F004BX-L150			Unit
		Min	Typ ⁽¹⁾	Max	
Boot/Parameter Block Erase Time	2		2.0	8.6	s
Main Block Erase Time	2		3.4	17.0	s
Main Block Byte Program Time	2		1.4	5.3	s
Main Block Word Program Time	2		0.7	2.7	s

NOTES:

1. 25°C, 12.0V V_{pp}.
2. Excludes System-Level Overhead.



290450-19

Figure 18. A.C. Waveforms for Write and Erase Operations (WE#-Controlled Writes)

A.C. CHARACTERISTICS FOR CE #-CONTROLLED WRITE OPERATIONS

 $V_{CC} = 3.3V \pm 0.3V, 5.0V \pm 10\%$

Versions				28F400BX-L150 28F004BX-L150		Unit
Symbol	Parameter		Notes	Min	Max	
t_{AVAV}	t_{WC}	Write Cycle Time		150		ns
t_{PHEL}	t_{PS}	RP # High Recovery to CE # Going Low		1.0		μs
t_{WLEL}	t_{WS}	WE # Setup to CE # Going Low		0		ns
t_{PHHEH}	t_{PHS}	RP # V_{HH} Setup to CE # Going High	6, 8	200		ns
t_{VPEH}	t_{VPS}	V_{PP} Setup to CE # Going High	5, 8	200		ns
t_{AVEH}	t_{AS}	Address Setup to CE # Going High	3	95		ns
t_{DVEH}	t_{DS}	Data Setup to CE # Going High	4	100		ns
t_{ELEH}	t_{CP}	CE # Pulse Width		100		ns
t_{EHDH}	t_{DH}	Data Hold from CE # High	4	0		ns
t_{EHAX}	t_{AH}	Address Hold from CE # High	3	10		ns
t_{EHHH}	t_{WH}	WE # Hold from CE # High		10		ns
t_{EHEL}	t_{CPH}	CE # Pulse Width High		50		ns
t_{EHQV1}		Duration of Programming Operation Word/Byte	2, 5, 6	6		μs
t_{EHQV2}		Duration of Erase Operation (Boot)	2, 5, 6	0.3		s
t_{EHQV3}		Duration of Erase Operation (Parameter)	2, 5, 6	0.3		s
t_{EHQV4}		Duration of Erase Operation (Main)	2, 5, 6	0.6		s
t_{QWL}	t_{VPH}	V_{PP} Hold from Valid SRD	5, 8	0		ns
t_{QVPH}	t_{PPH}	RP # V_{HH} Hold from Valid SRD	6, 8	0		ns
t_{PHBR}		Boot-Block Relock Delay	7, 8		200	ns

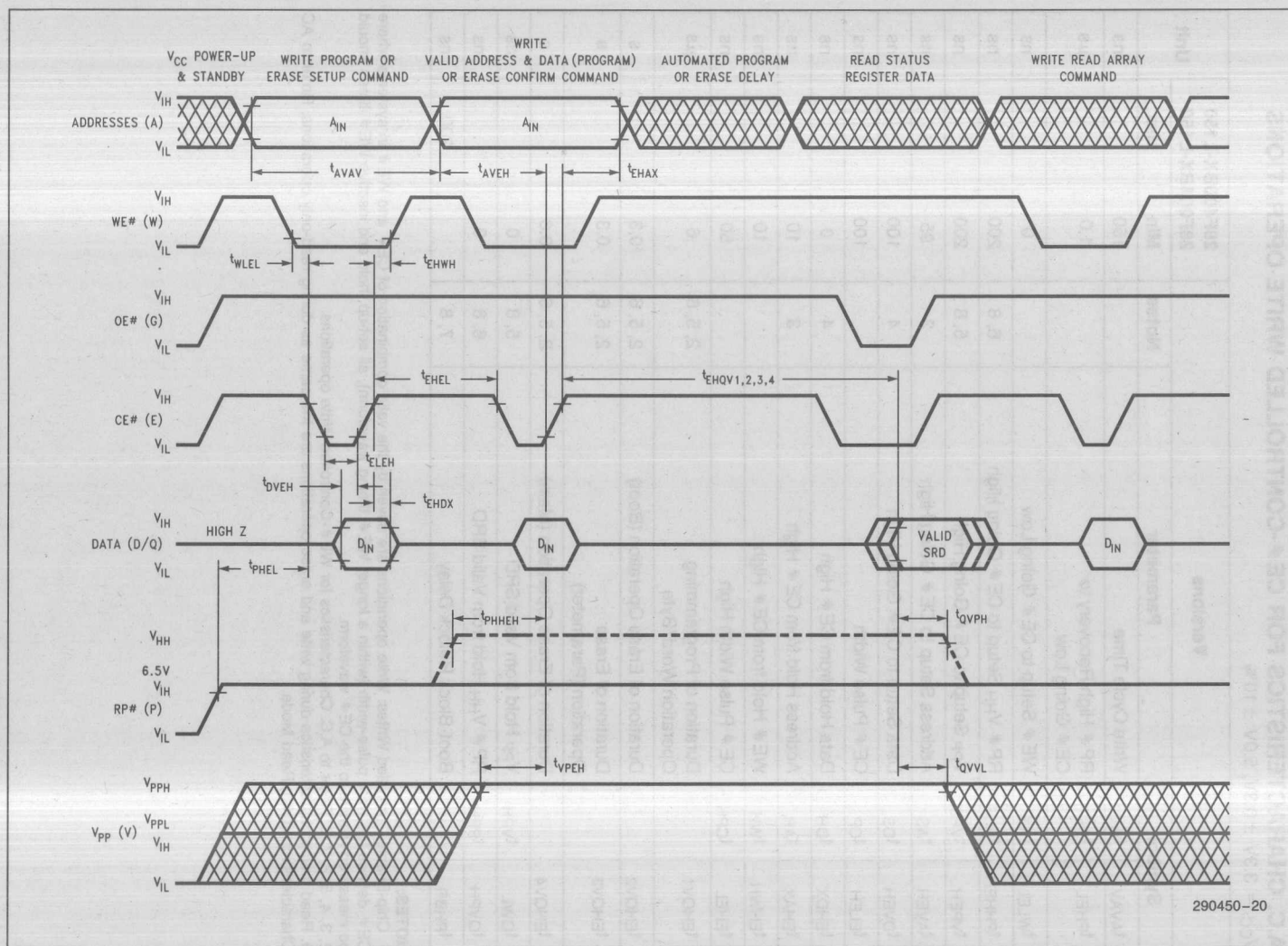
NOTES:

1. Chip-Enable Controlled Writes: Write operations are driven by the valid combination of CE # and WE # in systems where CE # defines the write pulse-width (within a longer WE # timing waveform), all set-up, hold and inactive WE # times should be measured relative to the CE # waveform.
- 2, 3, 4, 5, 6, 7, 8. Refer to A.C. Characteristics for WE #-Controlled Write operations.
9. Read timing characteristics during write and erase operations are the same as during read-only operations. Refer to AC Characteristics during Read Mode.

4-90

PRELIMINARY

Figure 19. Alternate A.C. Waveforms for Write and Erase Operations (CE #-Controlled Writes)



ORDERING INFORMATION

E	2	8	F	4	0	0	B	X	-	L	1	5	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---

PACKAGE
E = STANDARD 56 LEAD TSOP
PA = 44 LEAD PSOP

ACCESS SPEED (ns)
150 ns

290450-21

VALID COMBINATIONS:
E28F400BX-L150 PA28F400BX-L150

E 2 8 F 0 0 4 B X - L 1 5 0											
PACKAGE									ACCESS SPEED (ns)		
E = STANDARD 40 LEAD TSOP									150 ns		
290450-22											
VALID COMBINATIONS:											
E28F004BX-L150											

ADDITIONAL INFORMATION

28F200BX/28F002BX Data Sheet
28F200BXL/28F002BXL Data Sheet
28F400BX/28F004BX Data Sheet

AP-363 "Extended Flash BIOS Design for Portable Computers"
AP-357 "Power Supply Solutions for Flash Memory"
ER-28 "ETOX-III Flash Memory Technology"
ER-29 "The Intel 2/4-Mbit Boot Block Flash Memory Family"

Order Number

290448
290449
290451
292098
292092
294012
294013

4

REVISION HISTORY

Number	Description
-001	Original Version
-002	Modified BYTE# Timing Waveforms Modified t _{DVWH} parameter for AC Characteristics for Write Operations
-003	PWD renamed to RP# for JEDEC standardization compatibility. Combined V _{CC} Read Current for 28F400BX-L Word-Wide and Byte-Wide Mode and 28F004BX-L Byte Wide Mode in DC Characteristics tables. Changed I _{PPS} current spec from ±10 μA to ±15 μA in DC Characteristics table. Added Boot Block Unlock current spec in DC Characteristics tables. Improved t _{PWH} spec to 600 ns (was 700 ns). Changed I _{CCR} maximum spec from 20 mA to 25 mA, and added 15 mA typical spec in DC Characteristics Table.

28F200BX-T/B, 28F002BX-T/B 2-MBIT (128K x 16, 256K x 8) BOOT BLOCK FLASH MEMORY FAMILY

- | | |
|--|---|
| <ul style="list-style-type: none"> ■ x8/x16 Input/Output Architecture <ul style="list-style-type: none"> — 28F200BX-T, 28F200BX-B — For High Performance and High Integration 16-bit and 32-bit CPUs ■ x8-only Input/Output Architecture <ul style="list-style-type: none"> — 28F002BX-T 28F002BX-B — For Space Constrained 8-bit Applications ■ Optimized High Density Blocked Architecture <ul style="list-style-type: none"> — One 16-KB Protected Boot Block — Two 8-KB Parameter Blocks — One 96-KB Main Block — One 128 KB Main Block — Top or Bottom Boot Locations ■ Extended Cycling Capability <ul style="list-style-type: none"> — 100,000 Block Erase Cycles ■ Automated Word/Byte Write and Block Erase <ul style="list-style-type: none"> — Command User Interface — Status Registers — Erase Suspend Capability ■ SRAM-Compatible Write Interface ■ Automatic Power Savings Feature <ul style="list-style-type: none"> — 1 mA Typical I_{CC} Active Current in Static Operation ■ Hardware Data Protection Feature <ul style="list-style-type: none"> — Erase/Write Lockout during Power Transitions | <ul style="list-style-type: none"> ■ Very High-Performance Read <ul style="list-style-type: none"> — 60/80 ns Maximum Access Time — 30/40 ns Maximum Output Enable Time ■ Low Power Consumption <ul style="list-style-type: none"> — 20 mA Typical x8 Active Read Current — 25 mA Typical x16 Active Read Current ■ Reset/Deep Power-Down Input <ul style="list-style-type: none"> — 0.2 μA I_{CC} Typical — Acts as Reset for Boot Operations ■ Extended Temperature Operation <ul style="list-style-type: none"> — -40°C to +85°C ■ Write Protection for Boot Block ■ Industry Standard Surface Mount Packaging <ul style="list-style-type: none"> — 28F200BX: JEDEC ROM Compatible <ul style="list-style-type: none"> 44-Lead PSOP 56-Lead TSOP — 28F002BX: 40-Lead TSOP ■ 12V Word/Byte Write and Block Erase <ul style="list-style-type: none"> — $V_{PP} = 12V \pm 5\%$ Standard — $V_{PP} = 12V \pm 10\%$ Option ■ ETOX III Flash Technology <ul style="list-style-type: none"> — 5V Read ■ Independent Software Vendor Support |
|--|---|

Intel's 2-Mbit Flash Memory Family is an extension of the Boot Block Architecture which includes block-selective erasure, automated write and erase operations and standard microprocessor interface. The 2-Mbit Flash Memory Family enhances the Boot Block Architecture by adding more density and blocks, x8/x16 input/output control, very high speed, low power, an industry standard ROM compatible pinout and surface mount packaging. The 2-Mbit flash family allows for an easy upgrade to Intel's 4-Mbit Boot Block Flash Memory Family.

The Intel 28F200BX-T/B are 16-bit wide flash memory offerings. These high density flash memories provide user selectable bus operation for either 8-bit or 16-bit applications. The 28F200BX-T and 28F200BX-B are 2,097,152-bit non-volatile memories organized as either 262,144 bytes or 131,072 words of information. They are offered in 44-Lead plastic SOP and 56-Lead TSOP packages. The x8/x16 pinout conforms to the industry standard ROM/EPROM pinout.

The Intel 28F002BX-T/B are 8-bit wide flash memories with 2,097,152 bits organized as 262,144 bytes of information. They are offered in a 40-Lead TSOP package, which is ideal for space-constrained portable systems.

These devices use an integrated Command User Interface (CUI) and Write State Machine (WSM) for simplified word/byte write and block erasure. The 28F200BX-T/28F002BX-T provide block locations compatible with Intel's MCS-186 family, 80286, i386™, i486™, i860™ and 80960CA microprocessors. The 28F200BX-B/28F002BX-B provide compatibility with Intel's 80960KX and 80960SX families as well as other embedded microprocessors.

The boot block includes a data protection feature to protect the boot code in critical applications. With a maximum access time of 60 ns, these 2-Mbit flash devices are very high performance memories which interface at zero-wait-state to a wide range of microprocessors and microcontrollers. A deep power-down mode lowers the total V_{CC} power consumption to 1 μW typical. This is critical in handheld battery powered systems. For very low power applications using a 3.3V supply, refer to the Intel 28F200BX-TL/BL, 28F002BX-TL/BL 2-Mbit Boot Block Flash Memory Family datasheet.

Manufactured on Intel's 0.8 micron ETOXIII process, the 2-Mbit flash memory family provides world class quality, reliability and cost-effectiveness at the 2-Mbit density level.

1.0 PRODUCT FAMILY OVERVIEW

Throughout this datasheet the 28F200BX refers to both the 28F200BX-T and 28F200BX-B devices and 28F002BX refers to both the 28F002BX-T and 28F002BX-B devices. The 2-Mbit flash memory family refers to both the 28F200BX and 28F002BX products. This datasheet comprises the specifications for four separate products in the 2-Mbit flash memory family. Section 1 provides an overview of the 2-Mbit flash memory family including applications, pinouts and pin descriptions. Sections 2 and 3 describe in detail the specific memory organizations for the 28F200BX and 28F002BX products respectively. Section 4 combines a description of the family's principles of operations. Finally Section 5 describes the family's operating specifications.

PRODUCT FAMILY

x8/x16 Products	x8-Only Products
28F200BX-T	28F002BX-T
28F200BX-B	28F002BX-B

1.1 Main Features

The 28F200BX/28F002BX boot block flash memory family is a very high performance 2-Mbit (2,097,152 bit) memory family organized as either 128 KWords (131,072 words) of 16 bits each or 256 Kbytes (262,144 bytes) of 8 bits each.

Five Separately Erasable Blocks including a **hardware-lockable boot block** (16,384 Bytes), **two parameter blocks** (8,192 Bytes each) and **two main blocks** (1 block of 98,304 Bytes and 1 block of 131,072 Bytes) are included on the 2-Mbit family. An erase operation erases one of the main blocks in typically 2.4 seconds, and the boot or parameter blocks in typically 1.0 second. Each block can be independently erased and programmed 100,000 times.

The Boot Block is located at either the top (28F200BX-T, 28F002BX-T) or the bottom (28F200BX-B, 28F002BX-B) of the address map in order to accommodate different microprocessor protocols for boot code location. The **hardware lockable boot block** provides the most secure code storage. The boot block is intended to store the kernel code required for booting-up a system. When the **RP#** pin is between 11.4V and 12.6V the boot block is unlocked and program and erase operations can be performed. When the **RP#** pin is at or below 6.5V the boot block is locked and program and erase operations to the boot block are ignored.

The 28F200BX products are available in the ROM/EPROM compatible pinout and housed in the 44-Lead PSOP (Plastic Small Outline) package and the 56-Lead TSOP (Thin Small Outline, 1.2mm thick) package as shown in Figures 3 and 4. The 28F002BX products are available in the 40-Lead TSOP (1.2mm thick) package as shown in Figure 5.

The **Command User Interface (CUI)** serves as the interface between the microprocessor or microcontroller and the internal operation of the 28F200BX and 28F002BX flash memory products.

Program and Erase Automation allows program and erase operations to be executed using a two-write command sequence to the CUI. The internal Write State Machine (WSM) automatically executes the algorithms and timings necessary for program and erase operations, including verifications, thereby unburdening the microprocessor or microcontroller. Writing of memory data is performed in word or byte increments for the 28F200BX family and in byte increments for the 28F002BX family typically within 9 μ s which is a 100% improvement over current flash memory products.

The **Status Register (SR)** indicates the status of the WSM and whether the WSM successfully completed the desired program or erase operation.

Maximum Access Time of **60 ns (TACC)** is achieved over the commercial temperature range (0°C to 70°C), 5% V_{CC} supply voltage range (4.75V to 5.25V) and 30 pF output load. Refer to Figure 19; TACC vs Output Load Capacitance for larger output loads. Maximum Access Time of **80 ns (TACC)** is achieved over the commercial temperature range, 10% V_{CC} supply range (4.5V to 5.5V) and 100 pF output load.

Ipp maximum Program current is 40 mA for x16 operation and 30 mA for x8 operation. Ipp Erase current is 30 mA maximum. Vpp erase and programming voltage is 11.4V to 12.6V ($V_{pp} = 12V \pm 5\%$) under all operating conditions. As an option, V_{pp} can also vary between 10.8V to 13.2V ($V_{pp} = 12V \pm 10\%$) with a guaranteed number of 100 block erase cycles.

Typical I_{CC} Active Current of 25 mA is achieved for the x16 products (28F200BX), **typical I_{CC} Active Current of 20 mA** is achieved for the x8 products (28F200BX, 28F002BX). Refer to the I_{CC} active current derating curves in this datasheet.

The 2-Mbit boot block flash family is also designed with an Automatic Power Savings (APS) feature to minimize system battery current drain and allow for very low power designs. Once the device is ac-

cessed to read array data, APS mode will immediately put the memory in static mode of operation where I_{CC} active current is typically 1 mA until the next read is initiated.

When the CE# and RP# pins are at V_{CC} and the BYTE# pin (28F200BX-only) is at either V_{CC} or GND the **CMOS Standby** mode is enabled where I_{CC} is typically 50 μ A.

A **Deep Power-Down Mode** is enabled when the RP# pin is at ground minimizing power consumption and providing write protection during power-up conditions. **I_{CC} current** during deep power-down mode is **0.20 μ A typical**. An initial maximum access time or Reset Time of 300 ns is required from RP# switching until outputs are valid. Equivalently, the device has a maximum wake-up time of 215 ns until writes to the Command User Interface are recognized. When RP# is at ground the WSM is reset, the Status Register is cleared and the entire device is protected from being written to. This feature prevents data corruption and protects the code stored in the device during system reset. The system Reset pin can be tied to RP# to reset the memory to normal read mode upon activation of the Reset pin. With on-chip program/erase automation in the 2-Mbit family and the RP# functionality for data protection, when the CPU is reset and even if a program or erase command is issued, the device will not recognize any operation until RP# returns to its normal state.

For the 28F200BX, Byte-wide or Word-wide Input/Output Control is possible by controlling the BYTE# pin. When the BYTE# pin is at a logic low the device is in the byte-wide mode (x8) and data is read and written through DQ[0:7]. During the byte-wide mode, DQ[8:14] are tri-stated and DQ15/A-1 becomes the lowest order address pin. When the BYTE# pin is at a logic high the device is in the word-wide mode (x16) and data is read and written through DQ[0:15].

1.2 Applications

The 2-Mbit boot block flash family combines high density, high performance, cost-effective flash memories with blocking and hardware protection capabilities. Its flexibility and versatility will reduce costs throughout the product life cycle. Flash memory is ideal for Just-In-Time production flow, reducing system inventory and costs, and eliminating component handling during the production phase. During the product life cycle, when code updates or feature en-

hancements become necessary, flash memory will reduce the update costs by allowing either a user-performed code change via floppy disk or a remote code change via a serial link. The 2-Mbit boot block flash family provides full function, blocked flash memories suitable for a wide range of applications. These applications include **Extended PC BIOS**, **Digital Cellular Phone** program and data storage, **Telecommunication** boot/firmware, and various other embedded applications where both program and data storage are required.

Reprogrammable systems such as personal computers, are ideal applications for the 2-Mbit flash products. Portable and handheld personal computer applications are becoming more complex with the addition of power management software to take advantage of the latest microprocessor technology, the availability of ROM-based application software, pen tablet code for electronic hand writing, and diagnostic code. Figure 1 shows an example of a 28F200BX-T application.

This increase in software sophistication augments the probability that a code update will be required after the PC is shipped. The 2-Mbit flash products provide an inexpensive update solution for the notebook and handheld personal computers while extending their product lifetime. Furthermore, the 2-Mbit flash products' power-down mode provides added flexibility for these battery-operated portable designs which require operation at very low power levels.

The 2-Mbit flash products also provide excellent design solutions for Digital Cellular Phone and Telecommunication switching applications requiring high performance, high density storage capability coupled with modular software designs, and a small form factor package (x8-only bus). The 2-Mbit's blocking scheme allows for an easy segmentation of the embedded code with; 16 Kbytes of Hardware-Protected Boot code, 2 Main Blocks of program code and 2 Parameter Blocks of 8 Kbytes each for frequently updatable data storage and diagnostic messages (e.g. phone numbers, authorization codes). Figure 2 is an example of such an application with the 28F002BX-T.

These are a few actual examples of the wide range of applications for the 2-Mbit Boot Block flash memory family which enable system designers to achieve the best possible product design. Only your imagination limits the applicability of such a versatile product family.

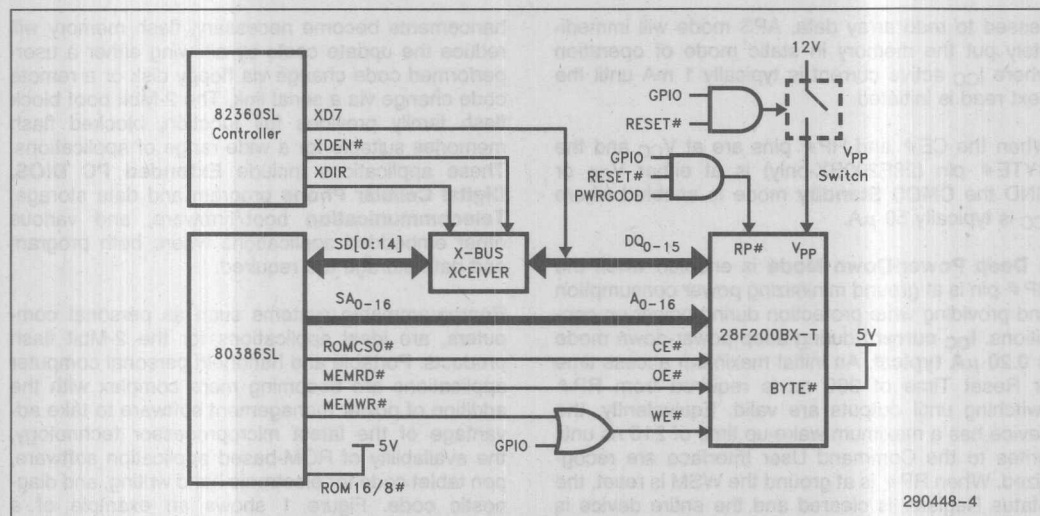


Figure 1. 28F200BX Interface to INTEL386SL Microprocessor Superset

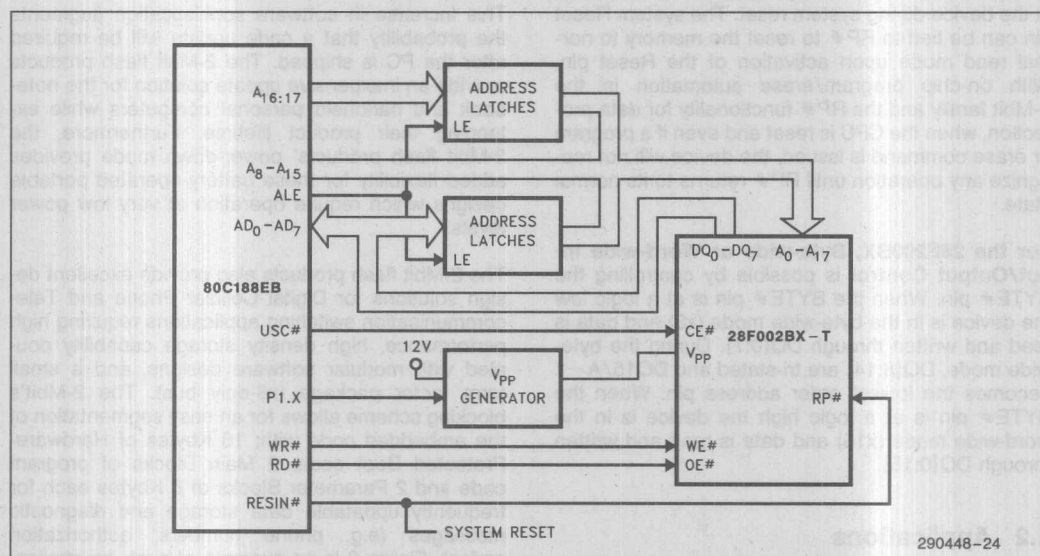


Figure 2. 28F002BX Interface to INTEL 80C188EB 8-Bit Embedded Microprocessor

1.3 Pinouts

The 28F200BX 44-Lead PSOP pinout follows the industry standard ROM/EPROM pinout as shown in Figure 3 with an upgrade to the 28F400BX (4-Mbit flash family). Furthermore, the 28F200BX 56-Lead TSOP pinout shown in Figure 4 provides density upgrades to the 28F400BX and to future higher density boot block memories.

The 28F002BX 40-Lead TSOP pinout shown in Figure 5 is 100% compatible and provides a density upgrade to the 28F004BX 4-Mbit Boot Block flash memory.

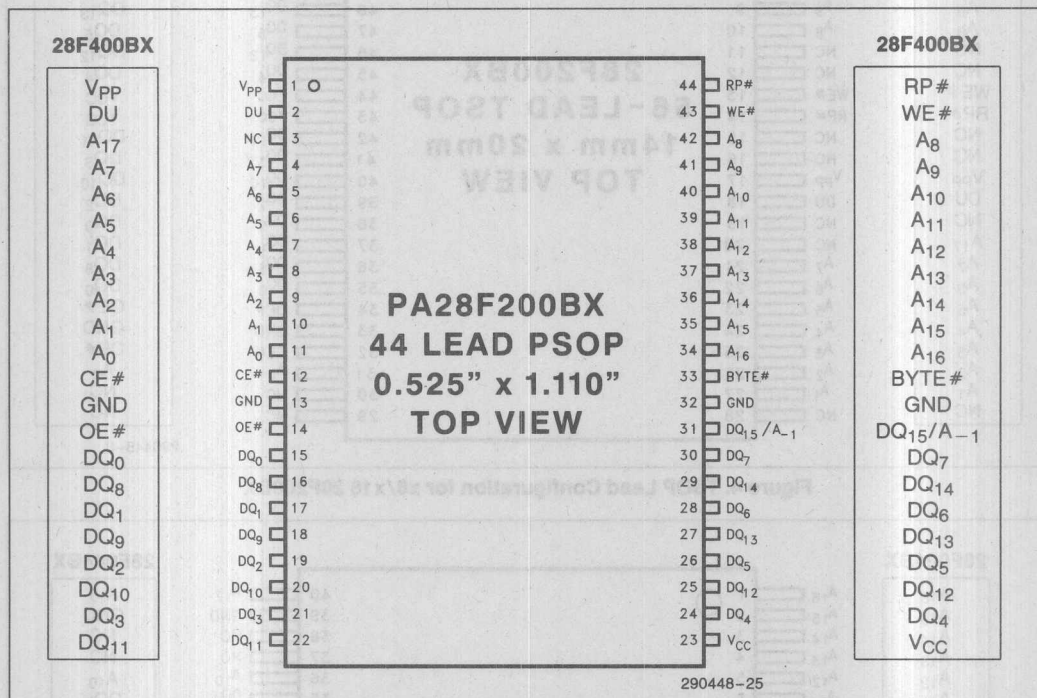


Figure 3. PSOP Lead Configuration for x8/x16 28F200BX

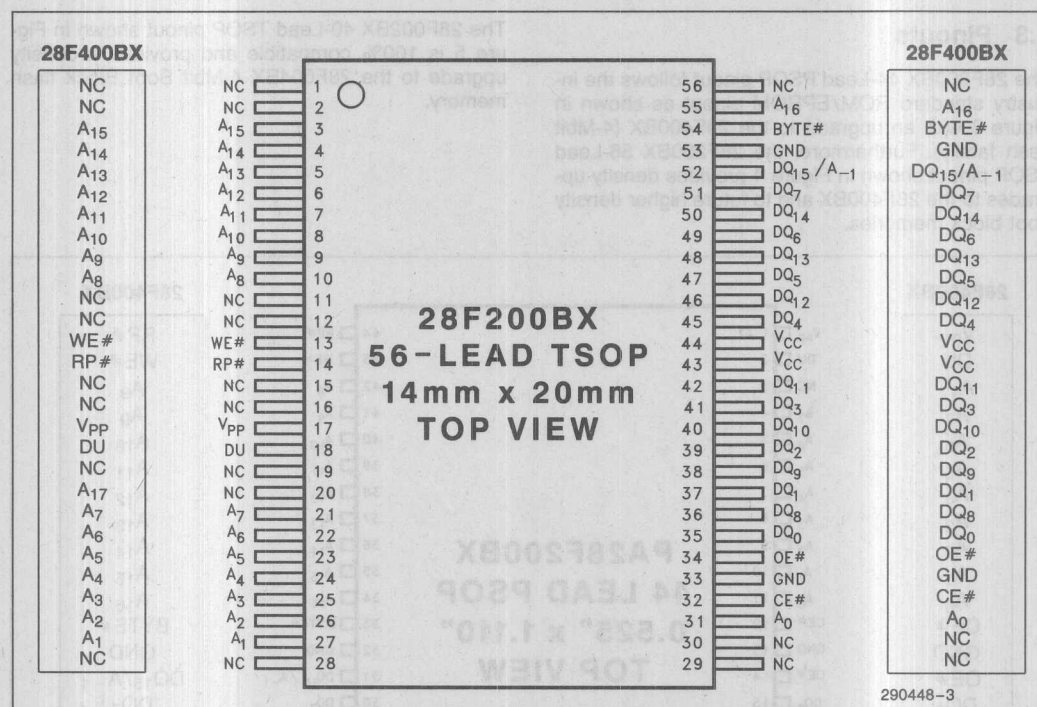


Figure 4. TSOP Lead Configuration for x8/x16 28F200BX

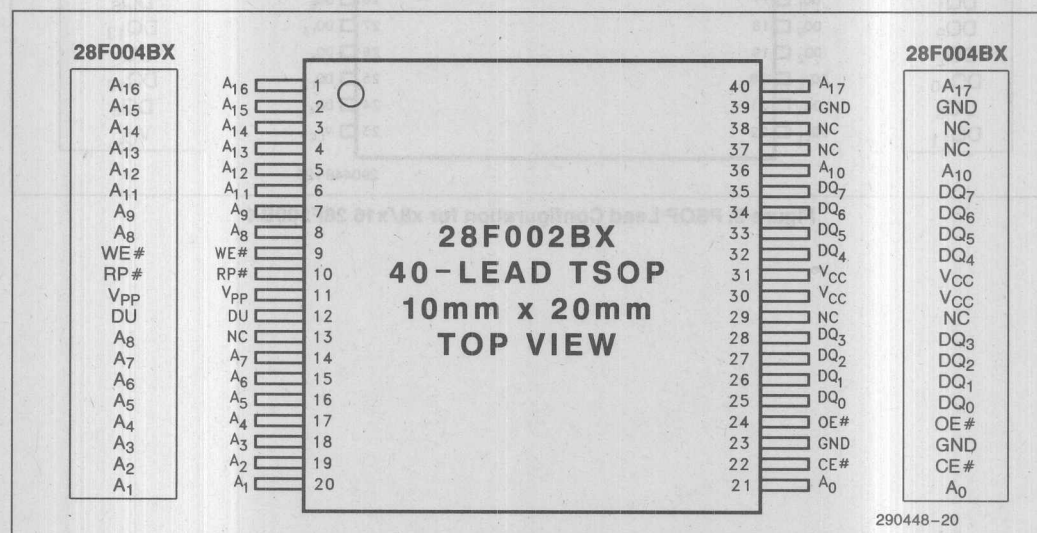


Figure 5. TSOP Lead Configuration for x8 28F002BX

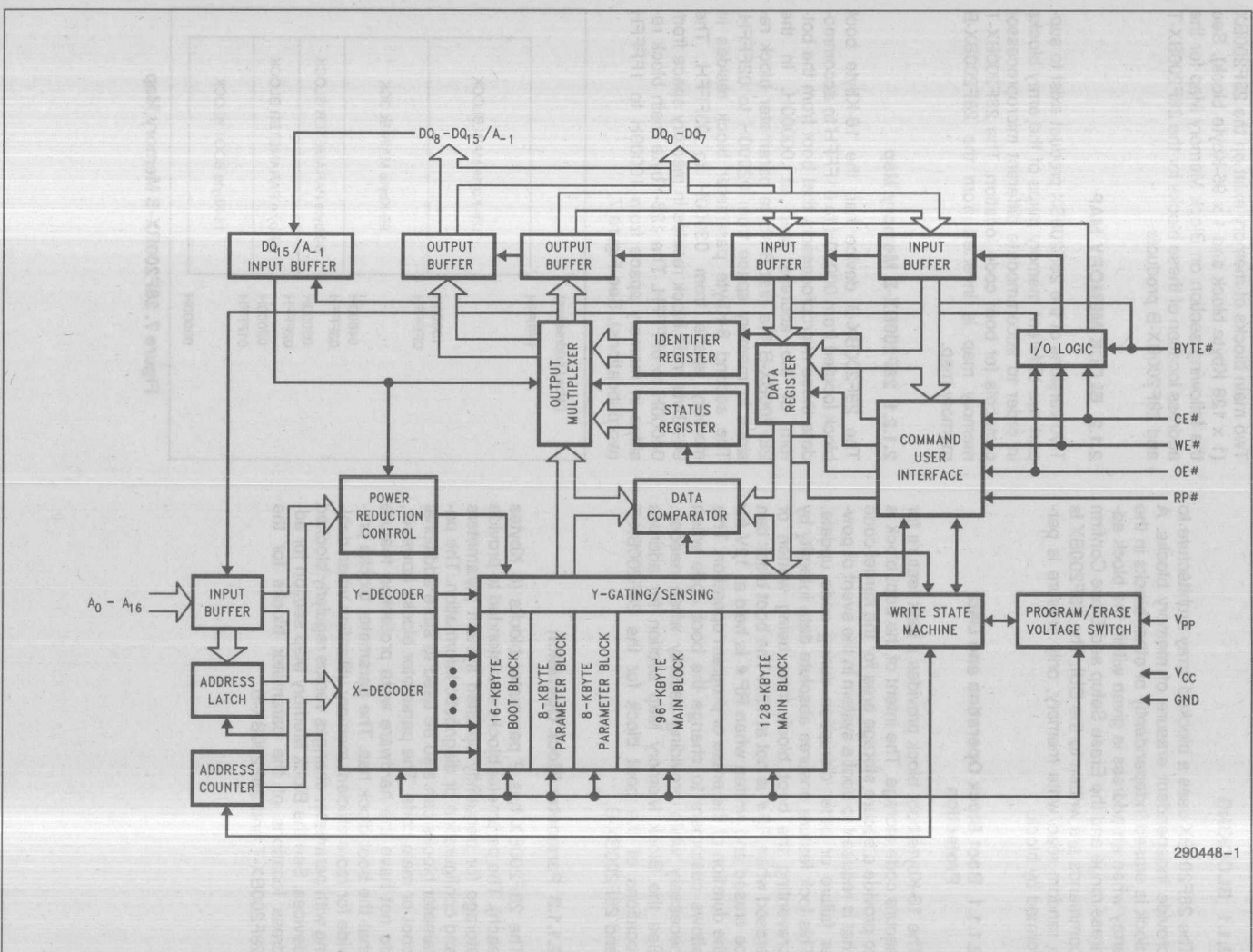
1.4 Pin Descriptions for the x8/x16 28F200BX

Symbol	Type	Name and Function
A ₀ -A ₁₆	I	ADDRESS INPUTS for memory addresses. Addresses are internally latched during a write cycle.
A ₉	I	ADDRESS INPUT: When A ₉ is at 12V the signature mode is accessed. During this mode A ₀ decodes between the manufacturer and device ID's. When BYTE # is at a logic low only the lower byte of the signatures are read. DQ ₁₅ /A ₋₁ is a don't care in the signature mode when BYTE # is low.
DQ ₀ -DQ ₇	I/O	DATA INPUTS/OUTPUTS: Inputs array data on the second CE # and WE # cycle during a program command. Inputs commands to the Command User Interface when CE # and WE # are active. Data is internally latched during the write and program cycles. Outputs array, Intelligent Identifier and Status Register data. The data pins float to tri-state when the chip is deselected or the outputs are disabled.
DQ ₈ -DQ ₁₅	I/O	DATA INPUTS/OUTPUTS: Inputs array data on the second CE # and WE # cycle during a program command. Data is internally latched during the write and program cycles. Outputs array data. The data pins float to tri-state when the chip is deselected or the outputs are disabled as in the byte-wide mode (BYTE # = "0"). In the byte-wide mode DQ ₁₅ /A ₋₁ becomes the lowest order address for data output on DQ ₀ -DQ ₇ .
CE #	I	CHIP ENABLE: Activates the device's control logic, input buffers, decoders and sense amplifiers. CE # is active low; CE # high deselects the memory device and reduces power consumption to standby levels. If CE # and RP # are high, but not at a CMOS high level, the standby current will increase due to current flow through the CE # and RP # input stages.
RP #	I	RESET/DEEP POWER-DOWN: Provides three-state control. Puts the device in deep power-down mode. Locks the boot block from program/erase. When RP # is at logic high level and equals 6.5V maximum the boot block is locked and cannot be programmed or erased. When RP # = 11.4V minimum the boot block is unlocked and can be programmed or erased. When RP # is at a logic low level the boot block is locked, the deep power-down mode is enabled and the WSM is reset preventing any blocks from being programmed or erased, therefore providing data protection during power transitions. When RP # transitions from logic low to logic high the flash memory enters the read array mode.
OE #	I	OUTPUT ENABLE: Gates the device's outputs through the data buffers during a read cycle. OE # is active low.
WE #	I	WRITE ENABLE: Controls writes to the Command Register and array blocks. WE # is active low. Addresses and data are latched on the rising edge of the WE # pulse.
BYTE #	I	BYTE # ENABLE: Controls whether the device operates in the byte-wide mode (x8) or the word-wide mode (x16). BYTE # pin must be controlled at CMOS levels to meet 100 μ A CMOS current in the standby mode. BYTE # = "0" enables the byte-wide mode, where data is read and programmed on DQ ₀ -DQ ₇ and DQ ₁₅ /A ₋₁ becomes the lowest order address that decodes between the upper and lower byte. DQ ₈ -DQ ₁₄ are tri-stated during the byte-wide mode. BYTE # = "1" enables the word-wide mode where data is read and programmed on DQ ₀ -DQ ₁₅ .
V _{PP}		PROGRAM/ERASE POWER SUPPLY: For erasing memory array blocks or programming data in each block. Note: V _{PP} < V _{PPLMAX} memory contents cannot be altered.
V _{CC}		DEVICE POWER SUPPLY (5V \pm 10%, 5V \pm 5%)
GND		GROUND: For all internal circuitry.
NC		NO CONNECT: Pin may be driven or left floating.
DU		DON'T USE PIN: Pin should not be connected to anything.

1.5 Pin Descriptions for x8 28F002BX

Symbol	Type	Name and Function
A ₀ –A ₁₇	I	ADDRESS INPUTS for memory addresses. Addresses are internally latched during a write cycle.
A ₉	I	ADDRESS INPUT: When A ₉ is at 12V the signature mode is accessed. During this mode A ₀ decodes between the manufacturer and device ID's.
DQ ₀ –DQ ₇	I/O	DATA INPUTS/OUTPUTS: Inputs array data on the second CE # and WE # cycle during a program command. Inputs commands to the command user interface when CE # and WE # are active. Data is internally latched during the write and program cycles. Outputs array, Intelligent Identifier and status register data. The data pins float to tri-state when the chip is deselected or the outputs are disabled.
CE #	I	CHIP ENABLE: Activates the device's control logic, input buffers, decoders and sense amplifiers. CE # is active low; CE # high deselects the memory device and reduces power consumption to standby levels. If CE # and RP # are high, but not at a CMOS high level, the standby current will increase due to current flow through the CE # and RP # input stages.
#RP #	I	RESET/DEEP POWERDOWN: Provides Three-State control. Puts the device in deep powerdown mode. Locks the Boot Block from program/erase. When RP # is at logic high level and equals 6.5V maximum the Boot Block is locked and cannot be programmed or erased. When RP # = 11.4V minimum the Boot Block is unlocked and can be programmed or erased. When RP # is at a logic low level the Boot Block is locked, the deep powerdown mode is enabled and the WSM is reset preventing any blocks from being programmed or erased, therefore providing data protection during power transitions. When RP # transitions from logic low to logic high, the flash memory enters the read-array mode.
OE #	I	OUTPUT ENABLE: Gates the device's outputs through the data buffers during a read cycle. OE # is active low.
WE #	I	WRITE ENABLE: Controls writes to the Command Register and array blocks. WE # is active low. Addresses and data are latched on the rising edge of the WE # pulse.
V _{PP}		PROGRAM/ERASE POWER SUPPLY: For erasing memory array blocks or programming data in each block. Note: V _{PP} < V _{PPLMAX} memory contents cannot be altered.
V _{CC}		DEVICE POWER SUPPLY (5V ± 10%, 5V ± 5%)
GND		GROUND: For all internal circuitry.
NC		NO CONNECT: Pin may be driven or left floating.
DU		DON'T USE PIN: Pin should not be connected to anything.

2.0 28F200BX WORD/BYTE-WIDE PRODUCTS DESCRIPTION



2.1 28F200BX Memory Organization

2.1.1 BLOCKING

The 28F200BX uses a blocked array architecture to provide independent erasure of memory blocks. A block is erased independently of other blocks in the array when an address is given within the block address range and the Erase Setup and Erase Confirm commands are written to the CUI. The 28F200BX is a random read/write memory, only erasure is performed by block.

2.1.1.1 Boot Block Operation and Data Protection

The 16-Kbyte boot block provides a lock feature for secure code storage. The intent of the boot block is to provide a secure storage area for the kernel code that is required to boot a system in the event of power failure or other disruption during code update. This lock feature ensures absolute data integrity by preventing the boot block from being written or erased when RP# is not at 12V. The boot block can be erased and written when RP# is held at 12V for the duration of the erase or program operation. This allows customers to change the boot code when necessary while providing security when needed. See the Block Memory Map section for address locations of the boot block for the 28F200BX-T and 28F200BX-B.

2.1.1.2 Parameter Block Operation

The 28F200BX has 2 parameter blocks (8 Kbytes each). The parameter blocks are intended to provide storage for frequently updated system parameters and configuration or diagnostic information. The parameter blocks can also be used to store additional boot or main code. The parameter blocks however, do not have the hardware write protection feature that the boot block has. The parameter blocks provide for more efficient memory utilization when dealing with parameter changes versus regularly blocked devices. See the Block Memory Map section for address locations of the parameter blocks for the 28F200BX-T and 28F200BX-B.

2.1.1.3 Main Block Operation

Two main blocks of memory exist on the 28F200BX (1 x 128 Kbyte block and 1 x 96-Kbyte block). See the following section on Block Memory Map for the address location of these blocks for the 28F200BX-T and 28F200BX-B products.

2.1.2 BLOCK MEMORY MAP

Two versions of the 28F200BX product exist to support two different memory maps of the array blocks in order to accommodate different microprocessor protocols for boot code location. The 28F200BX-T memory map is inverted from the 28F200BX-B memory map.

2.1.2.1 28F200BX-B Memory Map

The 28F200BX-B device has the 16-Kbyte boot block located from 00000H to 01FFFFH to accommodate those microprocessors that boot from the bottom of the address map, at 00000H. In the 28F200BX-B the first 8-Kbyte parameter block resides in memory space from 02000H to 02FFFFH. The second 8-Kbyte parameter block resides in memory space from 03000H to 03FFFFH. The 96-Kbyte main block resides in memory space from 04000H to 0FFFFH. The 128-Kbyte main block resides in memory space from 10000H to 1FFFFH (word locations). See Figure 7.

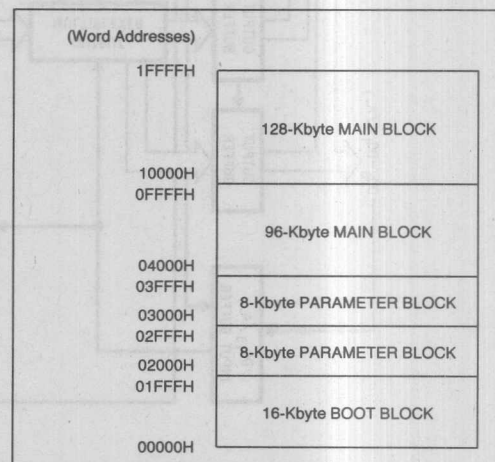


Figure 7. 28F200BX-B Memory Map

2.1.2.2 28F200BX-T Memory Map

The 28F200BX-T device has the 16-Kbyte boot block located from 1E000H to 1FFFFH to accommodate those microprocessors that boot from the top of the address map. In the 28F200BX-T the first 8-Kbyte parameter block resides in memory space from 1D000H to 1DFFFH. The second 8-Kbyte parameter block resides in memory space from 1C000H to 1CFFFH. The 96-Kbyte main block resides in memory space from 10000H to 1BFFFH. The 128-Kbyte main block resides in memory space from 00000H to 0FFFFH as shown in Figure 8.

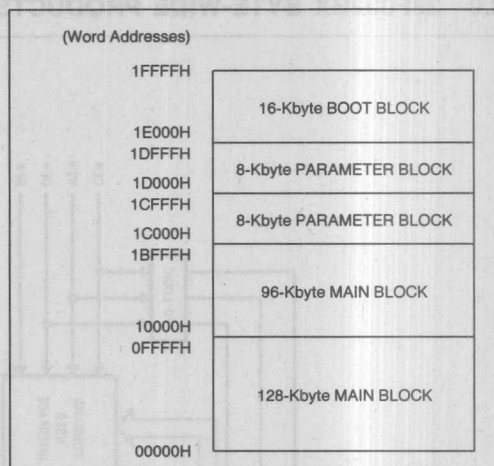
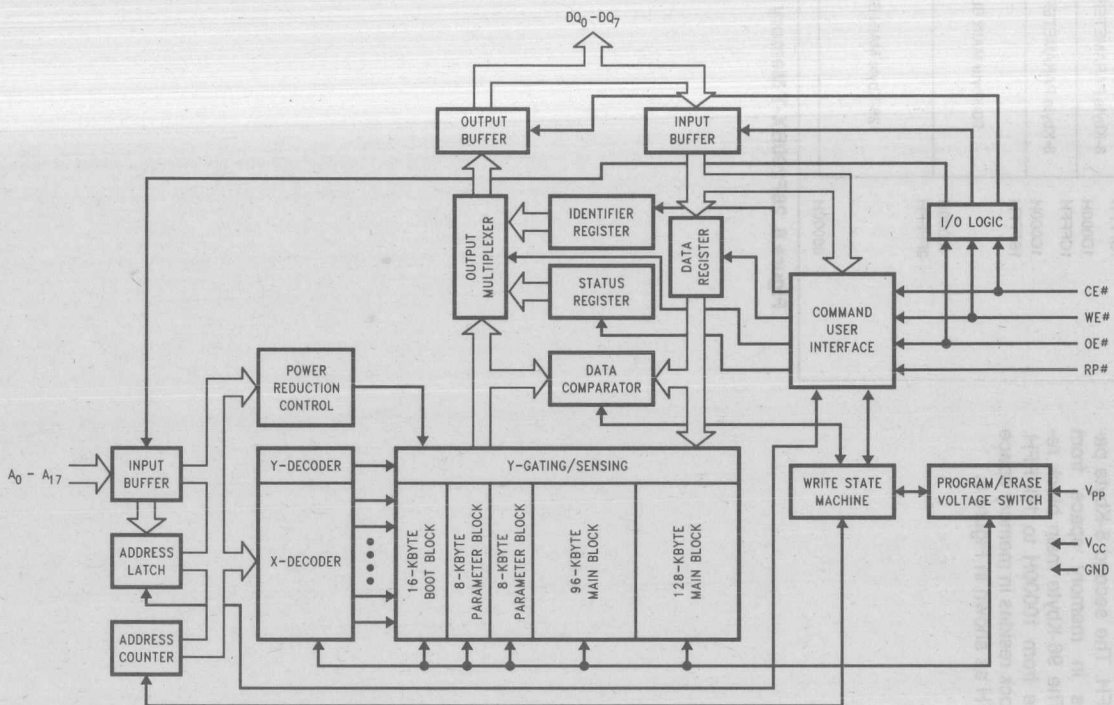


Figure 8. 28F200BX-T Memory Map

3.0 28F002BX BYTE-WIDE PRODUCTS DESCRIPTION



290448-19

Figure 9. 28F002BX Byte-Wide Block Diagram

3.1 28F002BX Memory Organization

3.1.1 BLOCKING

The 28F002BX uses a blocked array architecture to provide independent erasure of memory blocks. A block is erased independently of other blocks in the array when an address is given within the block address range and the Erase Setup and Erase Confirm commands are written to the CUI. The 28F002BX is a random read/write memory, only erasure is performed by block.

3.1.1.1 Boot Block Operation and Data Protection

The 16-Kbyte boot block provides a lock feature for secure code storage. The intent of the boot block is to provide a secure storage area for the kernel code that is required to boot a system in the event of power failure or other disruption during code update. This lock feature ensures absolute data integrity by preventing the boot block from being programmed or erased when RP# is not at 12V. The boot block can be erased and programmed when RP# is held at 12V for the duration of the erase or program operation. This allows customers to change the boot code when necessary while still providing security when needed. See the Block Memory Map section for address locations of the boot block for the 28F002BX-T and 28F002BX-B.

3.1.1.2 Parameter Block Operation

The 28F002BX has 2 parameter blocks (8 Kbytes each). The parameter blocks are intended to provide storage for frequently updated system parameters and configuration or diagnostic information. The parameter blocks can also be used to store additional boot or main code. The parameter blocks however, do not have the hardware write protection feature that the boot block has. Parameter blocks provide for more efficient memory utilization when dealing with small parameter changes versus regularly blocked devices. See the Block Memory Map section for address locations of the parameter blocks for the 28F002BX-T and 28F002BX-B.

3.1.1.3 Main Block Operation

Two main blocks of memory exist on the 28F002BX (1 x 128-Kbyte block and 1 x 96-Kbyte block). See the following section on Block Memory Map for address location of these blocks for the 28F002BX-T and 28F002BX-B.

3.1.2 BLOCK MEMORY MAP

Two versions of the 28F002BX product exist to support two different memory maps of the array blocks in order to accommodate different microprocessor protocols for boot code location. The 28F002BX-T memory map is inverted from the 28F002BX-B memory map.

3.1.2.1 28F002BX-B Memory Map

The 28F002BX-B device has the 16-Kbyte boot block located from 00000H to 03FFFFH to accommodate those microprocessors that boot from the bottom of the address map at 00000H. In the 28F002BX-B the first 8-Kbyte parameter block resides in memory from 04000H to 05FFFFH. The second 8-Kbyte parameter block resides in memory space from 06000H to 07FFFFH. The 96-Kbyte main block resides in memory space from 08000H to 1FFFFH. The 128-Kbyte main block resides in memory space from 20000H to 3FFFFH. See Figure 10.

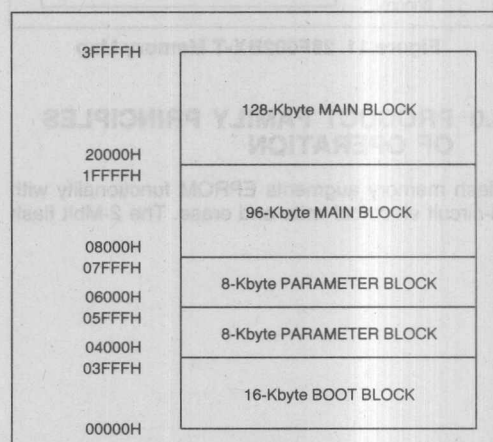


Figure 10. 28F002BX-B Memory Map

3.1.2.2 28F002BX-T Memory Map

The 28F002BX-T device has the 16-Kbyte boot block located from 3C000H to 3FFFFH to accommodate those microprocessors that boot from the top of the address map. In the 28F002BX-T the first 8-Kbyte parameter block resides in memory space from 3A000H to 3BFFFH. The second 8-Kbyte parameter block resides in memory space from 38000H to 39FFFH. The 96-Kbyte main block resides in memory space from 20000H to 37FFFH. The 128-Kbyte main block resides in memory space from 00000H to 1FFFFH.

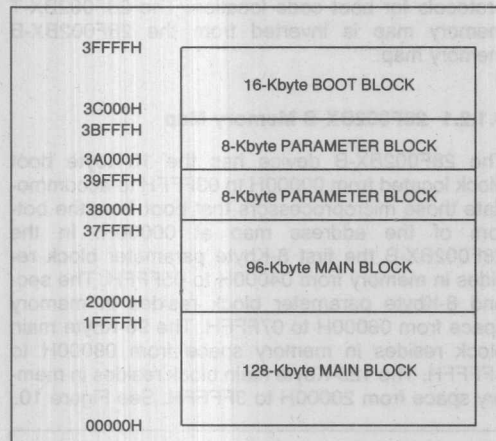


Figure 11. 28F002BX-T Memory Map

4.0 PRODUCT FAMILY PRINCIPLES OF OPERATION

Flash memory augments EPROM functionality with in-circuit electrical write and erase. The 2-Mbit flash

family utilizes a Command User Interface (CUI) and internally generated and timed algorithms to simplify write and erase operations.

The CUI allows for 100% TTL-level control inputs, fixed power supplies during erasure and programming, and maximum EPROM compatibility.

In the absence of high voltage on the V_{pp} pin, the 2-Mbit boot block flash family will only successfully execute the following commands: Read Array, Read Status Register, Clear Status Register and Intelligent Identifier mode. The device provides standard EPROM read, standby and output disable operations. Manufacturer Identification and Device Identification data can be accessed through the CUI or through the standard EPROM A9 high voltage access (V_{ID}) for PROM programming equipment.

The same EPROM read, standby and output disable functions are available when high voltage is applied to the V_{pp} pin. In addition, high voltage on V_{pp} allows write and erase of the device. All functions associated with altering memory contents: write and erase, Intelligent Identifier read and Read Status are accessed via the CUI.

The purpose of the Write State Machine (WSM) is to completely automate the write and erasure of the device. The WSM will begin operation upon receipt of a signal from the CUI and will report status back through a Status Register. The CUI will handle the WE# interface to the data and address latches, as well as system software requests for status while the WSM is in operation.

4.1 28F200BX Bus Operations

Flash memory reads, erases and writes in-system via the local CPU. All bus cycles to or from the flash memory conform to standard microprocessor bus cycles.

Table 1. Bus Operations for WORD-WIDE Mode (BYTE# = V_{IL})

Mode	Notes	RP#	CE#	OE#	WE#	A ₉	A ₀	V _{PP}	DQ ₀₋₁₅
Read	1, 2, 3	V _{IH}	V _{IL}	V _{IL}	V _{IH}	X	X	X	D _{OUT}
Output Disable		V _{IH}	V _{IL}	V _{IH}	V _{IH}	X	X	X	High Z
Standby		V _{IH}	V _{IH}	X	X	X	X	X	High Z
Deep Power-Down	9	V _{IL}	X	X	X	X	X	X	High Z
Intelligent Identifier (Mfr)	4	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IL}	X	0089H
Intelligent Identifier (Device)	4, 5	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IH}	X	2274H 2275H
Write	6, 7, 8	V _{IH}	V _{IL}	V _{IH}	V _{IL}	X	X	X	D _{IN}

Table 2. Bus Operations for BYTE-WIDE Mode (BYTE# = V_{IL})

Mode	Notes	RP#	CE#	OE#	WE#	A ₉	A ₀	A ₋₁	V _{PP}	DQ ₀₋₇	DQ ₈₋₁₄
Read	1, 2, 3	V _{IH}	V _{IL}	V _{IL}	V _{IH}	X	X	X	X	D _{OUT}	High Z
Output Disable		V _{IH}	V _{IL}	V _{IH}	V _{IH}	X	X	X	X	High Z	High Z
Standby		V _{IH}	V _{IH}	X	X	X	X	X	X	High Z	High Z
Deep Power-Down	9	V _{IL}	X	X	X	X	X	X	X	High Z	High Z
Intelligent Identifier (Mfr)	4	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IL}	X	X	89H	High Z
Intelligent Identifier (Device)	4, 5	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IH}	X	X	74H 75H	High Z
Write	6, 7, 8	V _{IH}	V _{IL}	V _{IH}	V _{IL}	X	X	X	X	D _{IN}	High Z

NOTES:

1. Refer to DC Characteristics.
2. X can be V_{IL} or V_{IH} for control pins and addresses, V_{PPL} or V_{PPH} for V_{PP}.
3. See DC characteristics for V_{PPL}, V_{PPH}, V_{HH}, V_{ID} voltages.
4. Manufacturer and Device codes may also be accessed via a CUI write sequence. A_{1-A17} = X.
5. Device ID = 2274H for 28F200BX-T and 2275H for 28F200BX-B.
6. Refer to Table 4 for valid D_{IN} during a write operation.
7. Command writes for Block Erase or Word/Byte Write are only executed when V_{PP} = V_{PPH}.
8. To write or erase the boot block, hold RP# at V_{IH}.
9. RP# must be at GND ±0.2V to meet the 1.2 μA maximum deep power-down current.

4.2 28F002BX Bus Operations

Table 3. Bus Operations

Mode	Notes	RP #	CE #	OE #	WE #	A ₉	A ₀	V _{PP}	DQ ₀₋₇
Read	1, 2, 3	V _{IH}	V _{IL}	V _{IL}	V _{IH}	X	X	X	D _{OUT}
Output Disable		V _{IH}	V _{IL}	V _{IH}	V _{IH}	X	X	X	High Z
Standby		V _{IH}	V _{IH}	X	X	X	X	X	High Z
Deep Power-Down	9	V _{IL}	X	X	X	X	X	X	High Z
Intelligent Identifier (Mfr)	4	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IL}	X	89H
Intelligent Identifier (Device)	4, 5	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IH}	X	7CH 7DH
Write	6, 7, 8	V _{IH}	V _{IL}	V _{IH}	V _{IL}	X	X	X	D _{IN}

NOTES:

1. Refer to DC Characteristics.
2. X can be V_{IL} or V_{IH} for control pins and addresses, V_{PPL} or V_{PPH} for V_{PP}.
3. See DC characteristics for V_{PPL}, V_{PPH}, V_{HH}, V_{ID} voltages.
4. Manufacturer and Device codes may also be accessed via a CUI write sequence. A₁-A₁₆ = X.
5. Device ID = 7CH for 28F002BX-T and 7DH for 28F002BX-B.
6. Refer to Table 4 for valid D_{IN} during a write operation.
7. Command writes for Block Erase or byte program are only executed when V_{PP} = V_{PPH}.
8. Program or erase the Boot block by holding RP# at V_{HH}.
9. RP# must be at GND ±0.2V to meet the 1.2 µA maximum deep power-down current.

4.3 Read Operations

The 2-Mbit boot block flash family has three user read modes; Array, Intelligent Identifier, and Status Register. Status Register read mode will be discussed in detail in the "Write Operations" section.

During power-up conditions (V_{CC} supply ramping), it takes a maximum of 600 ns from when V_{CC} is at 4.5V minimum to valid data on the outputs.

4.3.1 READ ARRAY

If the memory is not in the Read Array mode, it is necessary to write the appropriate read mode command to the CUI. The 2-Mbit boot block flash family has three control functions, all of which must be logically active, to obtain data at the outputs. Chip-Enable CE# is the device selection control. Power-Down RP# is the device power control. Output-Enable OE# is the DATA INPUT/OUTPUT (DQ[0:15] or DQ[0:7]) direction control and when active is used to drive data from the selected memory on to the I/O bus.

4.3.1.1 Output Control

With OE# at logic-high level (V_{IH}), the output from the device is disabled and data input/output pins

(DQ[0:15] or DQ[0:7]) are tri-stated. Data input is then controlled by WE#.

4.3.1.2 Input Control

With WE# at logic-high level (V_{IH}), input to the device is disabled. Data Input/Output pins (DQ[0:15] or DQ[0:7]) are controlled by OE#.

4.3.2 INTELLIGENT IDENTIFIERS

28F200BX Products

The manufacturer and device codes are read via the CUI or by taking the A₉ pin to 12V. Writing 90H to the CUI places the device into Intelligent Identifier read mode. A read of location 00000H outputs the manufacturer's identification code, 0089H, and location 00001H outputs the device code; 2274H for 28F200BX-T, 2275H for 28F200BX-B. When BYTE# is at a logic low only the lower byte of the above signatures is read and DQ₁₅/A₋₁ is a "don't care" during Intelligent Identifier mode. A read array command must be written to the CUI to return to the read array mode.



28F002BX Products

The manufacturer and device codes are also read via the CUI or by taking the A9 pin to 12V. Writing 90H to the CUI places the device into Intelligent Identifier read mode. A read of location 00000H outputs the manufacturer's identification code, 89H, and location 00001H outputs the device code; 7CH for 28F002BX-T, 7DH for 28F002BX-B.

4.4 Write Operations

Commands are written to the CUI using standard microprocessor write timings. The CUI serves as the interface between the microprocessor and the internal chip operation. The CUI can decipher Read Array, Read Intelligent Identifier, Read Status Register, Clear Status Register, Erase and Program commands. In the event of a read command, the CUI simply points the read path at either the array, the intelligent identifier, or the status register depending on the specific read command given. For a program or erase cycle, the CUI informs the write state machine that a write or erase has been requested. During a program cycle, the Write State Machine will control the program sequences and the CUI will only respond to status reads. During an erase cycle, the CUI will respond to status reads and erase suspend. After the Write State Machine has completed its task, it will allow the CUI to respond to its full command set. The CUI will stay in the current command state until the microprocessor issues another command.

The CUI will successfully initiate an erase or write operation only when V_{PP} is within its voltage range. Depending upon the application, the system designer may choose to make the V_{PP} power supply switchable, available only when memory updates are desired. The system designer can also choose to "hard-wire" V_{PP} to 12V. The 2 Mbit boot block flash family is designed to accommodate either design practice. It is recommended that RP# be tied to logical Reset for data protection during unstable CPU reset function as described in the "Product Family Overview" section.

4.4.1 BOOT BLOCK WRITE OPERATIONS

In the case of Boot Block modifications (write and erase), RP# is set to $V_{HH} = 12V$ typically, in addition to V_{PP} at high voltage. However, if RP# is not at V_{HH} when a program or erase operation of the boot block is attempted, the corresponding status register bit (Bit 4 for Program and Bit 5 for Erase, refer to Table 5 for Status Register Definitions) is set to indicate the failure to complete the operation.

4.4.2 COMMAND USER INTERFACE (CUI)

The Command User Interface (CUI) serves as the interface to the microprocessor. The CUI points the read/write path to the appropriate circuit block as described in the previous section. After the WSM has completed its task, it will set the WSM Status bit to a "1", which will also allow the CUI to respond to its full command set. Note that after the WSM has returned control to the CUI, the CUI will remain in its current state.

4.4.2.1 Command Set

Command Codes	Device Mode
00	Invalid/Reserved
10	Alternate Program Setup
20	Erase Setup
40	Program Setup
50	Clear Status Register
70	Read Status Register
90	Intelligent Identifier
B0	Erase Suspend
D0	Erase Resume/Erase Confirm
FF	Read Array

4.4.2.2 Command Function Descriptions

Device operations are selected by writing specific commands into the CUI. Table 4 defines the 2-Mbit boot block flash family commands.

Table 4. Command Definitions

Command	Bus Cycles Req'd	Notes	First Bus Cycle			Second Bus Cycle		
		8	Operation	Address	Data	Operation	Address	Data
Read Array/Reset	1	1	Write	X	FFH			
Intelligent Identifier	3	2, 4	Write	X	90H	Read	IA	IID
Read Status Register	2	3	Write	X	70H	Read	X	SRD
Clear Status Register	1		Write	X	50H			
Erase Setup/Erase Confirm	2	5	Write	BA	20H	Write	BA	D0H
Word/Byte Write Setup/Write	2	6, 7	Write	WA	40H	Write	WA	WD
Erase Suspend/Erase Resume	2		Write	X	B0H	Write	X	D0H
Alternate Word/Byte Write Setup/Write	2	6, 7	Write	WA	10H	Write	WA	WD

NOTES:

1. Bus operations are defined in Tables 1, 2, 3.
 2. IA = Identifier Address: 00H for manufacturer code, 01H for device code.
 3. SRD = Data read from Status Register.
 4. IID = Intelligent Identifier Data.
- Following the Intelligent Identifier Command, two read operations access manufacturer and device codes.
5. BA = Address within the block being erased.
 6. PA = Address to be programmed.
 7. PD = Data to be programmed at location PA.
 8. Either 40H or 10H command is valid.
 9. When writing commands to the device, the upper data bus [DQ8–DQ15] = X (28F200BX-only) which is either V_{CC} or V_{SS} to avoid burning additional current.

Invalid/Reserved

These are unassigned commands. It is not recommended that the customer use any command other than the valid commands specified above. Intel reserves the right to redefine these codes for future functions.

Read Array (FFH)

This single write command points the read path at the array. If the host CPU performs a CE#/OE# controlled read immediately following a two-write sequence that started the WSM, then the device will output status register contents. If the Read Array command is given after Erase Setup the device is reset to read the array. A two Read Array command sequence (FFH) is required to reset to Read Array after Program Setup.

Intelligent Identifier (90H)

After this command is executed, the CUI points the output path to the Intelligent Identifier circuits. Only Intelligent Identifier values at addresses 0 and 1 can be read (only address A0 is used in this mode, all other address inputs are ignored).

Read Status Register (70H)

This is one of the two commands that is executable while the state machine is operating. After this command is written, a read of the device will output the contents of the status register, regardless of the address presented to the device.

The device automatically enters this mode after program or erase has completed.

Clear Status Register (50H)

The WSM can only set the Program Status and Erase Status bits in the status register, it can not clear them. Two reasons exist for operating the status register in this fashion. The first is a synchronization. The WSM does not know when the host CPU has read the status register, therefore it would not know when to clear the status bits. Secondly, if the CPU is programming a string of bytes, it may be more efficient to query the status register after programming the string. Thus, if any errors exist while programming the string, the status register will return the accumulated error status.



Program Setup (40H or 10H)

This command simply sets the CUI into a state such that the next write will load the address and data registers. Either 40H or 10H can be used for Program Setup. Both commands are included to accommodate efforts to achieve an industry standard command code set.

Program

The second write after the program setup command, will latch addresses and data. Also, the CUI initiates the WSM to begin execution of the program algorithm. While the WSM finishes the algorithm, the device will output Status Register contents. Note that the WSM cannot be suspended during programming.

Erase Setup (20H)

Prepares the CUI for the Erase Confirm command. No other action is taken. If the next command is not an Erase Confirm command then the CUI will set both the Program Status and Erase Status bits of the Status Register to a "1", place the device into the Read Status Register state, and wait for another command.

Erase Confirm (D0H)

If the previous command was an Erase Setup command, then the CUI will enable the WSM to erase, at the same time closing the address and data latches, and respond only to the Read Status Register and Erase Suspend commands. While the WSM is executing, the device will output Status Register data when OE# is toggled low. Status Register data can only be updated by toggling either OE# or CE# low.

Erase Suspend (B0H)

This command only has meaning while the WSM is executing an Erase operation, and therefore will only be responded to during an erase operation. After this command has been executed, the CUI will set an output that directs the WSM to suspend Erase operations, and then return to responding to only Read Status Register or to the Erase Resume commands. Once the WSM has reached the Suspend state, it will set an output into the CUI which allows the CUI to respond to the Read Array, Read Status Register, and Erase Resume commands. In this mode, the CUI will not respond to any other commands. The WSM will also set the WSM Status bit to a "1". The WSM will continue to run, idling in the SUSPEND state, regardless of the state of all input

control pins, with the exclusion of RP#. RP# will immediately shut down the WSM and the remainder of the chip. During a suspend operation, the data and address latches will remain closed, but the address pads are able to drive the address into the read path.

Erase Resume (D0H)

This command will cause the CUI to clear the Suspend state and set the WSM Status bit to a "0", but only if an Erase Suspend command was previously issued. Erase Resume will not have any effect in all other conditions.

4.4.3 STATUS REGISTER

The 2-Mbit boot block flash family contains a status register which may be read to determine when a program or erase operation is complete, and whether that operation completed successfully. The status register may be read at any time by writing the Read Status command to the CUI. After writing this command, all subsequent Read operations output data from the status register until another command is written to the CUI. A Read Array command must be written to the CUI to return to the Read Array mode.

The status register bits are output on DQ[0:7] whether the device is in the byte-wide (x8) or word-wide (x16) mode for the 28F200BX. In the word-wide mode the upper byte, DQ[8:15] is set to 00H during a Read Status command. In the byte-wide mode, DQ[8:14] are tri-stated and DQ15/A-1 retains the low order address function.

It should be noted that the contents of the status register are latched on the falling edge of OE# or CE# whichever occurs last in the read cycle. This prevents possible bus errors which might occur if the contents of the status register change while reading the status register. CE# or OE# must be toggled with each subsequent status read, or the completion of a program or erase operation will not be evident.

The Status Register is the interface between the microprocessor and the Write State Machine (WSM). When the WSM is active, this register will indicate the status of the WSM, and will also hold the bits indicating whether or not the WSM was successful in performing the desired operation. The WSM sets status bits "Three" through "Seven" and clears bits "Six" and "Seven", but cannot clear status bits "Three" through "Five". These bits can only be cleared by the controlling CPU through the use of the Clear Status Register command.

4.4.3.1 Status Register Bit Definition

Table 5. Status Register Definitions

WSMS	ESS	ES	PS	VPPS	R	R	R
7	6	5	4	3	2	1	0

NOTES:

SR.7 = WRITE STATE MACHINE STATUS

- 1 = Ready
- 0 = Busy

SR.6 = ERASE SUSPEND STATUS

- 1 = Erase Suspended
- 0 = Erase in Progress/Completed

SR.5 = ERASE STATUS

- 1 = Error in Block Erasure
- 0 = Successful Block Erase

SR.4 = PROGRAM STATUS

- 1 = Error in Byte/Word Program
- 0 = Successful Byte/Word Program

SR.3 = Vpp STATUS

- 1 = Vpp Low Detect; Operation Abort
- 0 = Vpp OK

SR.2-SR.0 = RESERVED FOR FUTURE ENHANCEMENTS

Write State Machine Status bit must first be checked to determine byte/word program or block erase completion, before the Program or Erase Status bits are checked for success.

When Erase Suspend is issued, WSM halts execution and sets both WSMS and ESS bits to "1". ESS bit remains set to "1" until an Erase Resume command is issued.

When this bit is set to "1", WSM has applied the maximum number of erase pulses to the block and is still unable to successfully perform an erase verify.

When this bit is set to "1", WSM has attempted but failed to Program a byte or word.

The Vpp Status bit, unlike an A/D converter, does not provide continuous indication of Vpp level. The WSM interrogates the Vpp level only after the byte write or block erase command sequences have been entered and informs the system if Vpp has not been switched on. The Vpp Status bit is not guaranteed to report accurate feedback between VppL and VppH.

These bits are reserved for future use and should be masked out when polling the Status Register.

4.4.3.2 Clearing the Status Register

Certain bits in the status register are set by the write state machine, and can only be reset by the system software. These bits can indicate various failure conditions. By allowing the system software to control the resetting of these bits, several operations may be performed (such as cumulatively programming several bytes or erasing multiple blocks in sequence). The status register may then be read to determine if an error occurred during that programming or erasure series. This adds flexibility to the way the device may be programmed or erased. To clear the status register, the Clear Status Register command is written to the CUI. Then, any other command may be issued to the CUI. Note again that before a read cycle can be initiated, a Read Array command must be written to the CUI to specify whether the read data is to come from the array, status register, or Intelligent Identifier.

4.4.4 PROGRAM MODE

Program is executed by a two-write sequence. The Program Setup command is written to the CUI followed by a second write which specifies the address and data to be programmed. The write state machine will execute a sequence of internally timed events to:

1. Program the desired bits of the addressed memory word (byte), and
2. Verify that the desired bits are sufficiently programmed

Programming of the memory results in specific bits within a byte or word being changed to a "0".

If the user attempts to program "1"s, there will be no change of the memory cell content and no error occurs.

Similar to erasure, the status register indicates whether programming is complete. While the program sequence is executing, bit 7 of the status register is a "0". The status register can be polled by

toggling either CE# or OE# to determine when the program sequence is complete. Only the Read Status Register command is valid while programming is active.

When programming is complete, the status bits, which indicate whether the program operation was successful, should be checked. If the programming operation was unsuccessful, Bit 4 of the status register is set to a "1" to indicate a Program Failure. If Bit 3 is set then V_{pp} was not within acceptable limits, and the WSM will not execute the programming sequence.

The status register should be cleared before attempting the next operation. Any CUI instruction can follow after programming is completed; however, it must be recognized that reads from the memory, status register, or Intelligent Identifier cannot be accomplished until the CUI is given the appropriate command. A Read Array command must first be given before memory contents can be read.

Figure 12 shows a system software flowchart for device byte programming operation. Figure 13 shows a similar flowchart for device word programming operation (28F200BX-only).

4.4.5 ERASE MODE

Erasure of a single block is initiated by writing the Erase Setup and Erase Confirm commands to the CUI, along with the addresses, A[12:16] for the 28F200BX or A[12:17] for the 28F002BX, identifying the block to be erased. These addresses are latched internally when the Erase Confirm command is issued. Block erasure results in all bits within the block being set to "1".

The WSM will execute a sequence of internally timed events to:

1. Program all bits within the block
2. Verify that all bits within the block are sufficiently programmed
3. Erase all bits within the block and
4. Verify that all bits within the block are sufficiently erased

While the erase sequence is executing, Bit 7 of the status register is a "0".

When the status register indicates that erasure is complete, the status bits, which indicate whether the erase operation was successful, should be checked. If the erasure operation was unsuccessful, Bit 5 of the status register is set to a "1" to indicate an Erase Failure. If V_{pp} was not within acceptable limits after the Erase Confirm command is issued, the WSM will not execute an erase sequence; instead, Bit 5 of the status register is set to a "1" to indicate

an Erase Failure, and Bit 3 is set to a "1" to identify that V_{pp} supply voltage was not within acceptable limits.

The status register should be cleared before attempting the next operation. Any CUI instruction can follow after erasure is completed; however, it must be recognized that reads from the memory array, status register, or Intelligent Identifier can not be accomplished until the CUI is given the appropriate command. A Read Array command must first be given before memory contents can be read.

Figure 14 shows a system software flowchart for Block Erase operation.

4.4.5.1 Suspending and Resuming Erase

Since an erase operation typically requires 1 to 3 seconds to complete, an Erase Suspend command is provided. This allows erase-sequence interruption in order to read data from another block of the memory. Once the erase sequence is started, writing the Erase Suspend command to the CUI requests that the Write State Machine (WSM) pause the erase sequence at a predetermined point in the erase algorithm. The status register must be read to determine when the erase operation has been suspended.

At this point, a Read Array command can be written to the CUI in order to read data from blocks other than that which is being suspended. The only other valid command at this time is the Erase Resume command or Read Status Register operation.

Figure 15 shows a system software flowchart detailing the operation.

During Erase Suspend mode, the chip can go into a pseudo-standby mode by taking CE# to V_{IH} and the active current is now a maximum of 10 mA. If the chip is enabled while in this mode by taking CE# to V_{IL}, the Erase Resume command can be issued to resume the erase operation.

Upon completion of reads from any block other than the block being erased, the Erase Resume command must be issued. When the Erase Resume command is given, the WSM will continue with the erase sequence and complete erasing the block. As with the end of erase, the status register must be read, cleared, and the next instruction issued in order to continue.

4.4.6 EXTENDED CYCLING

Intel has designed extended cycling capability into its ETOX III flash memory technology. The 2-Mbit boot block flash family is designed for 100,000 program/erase cycles on each of the five blocks. The combination of low electric fields, clean oxide processing and minimized oxide area per memory cell subjected to the tunneling electric field, results in very high cycling capability.

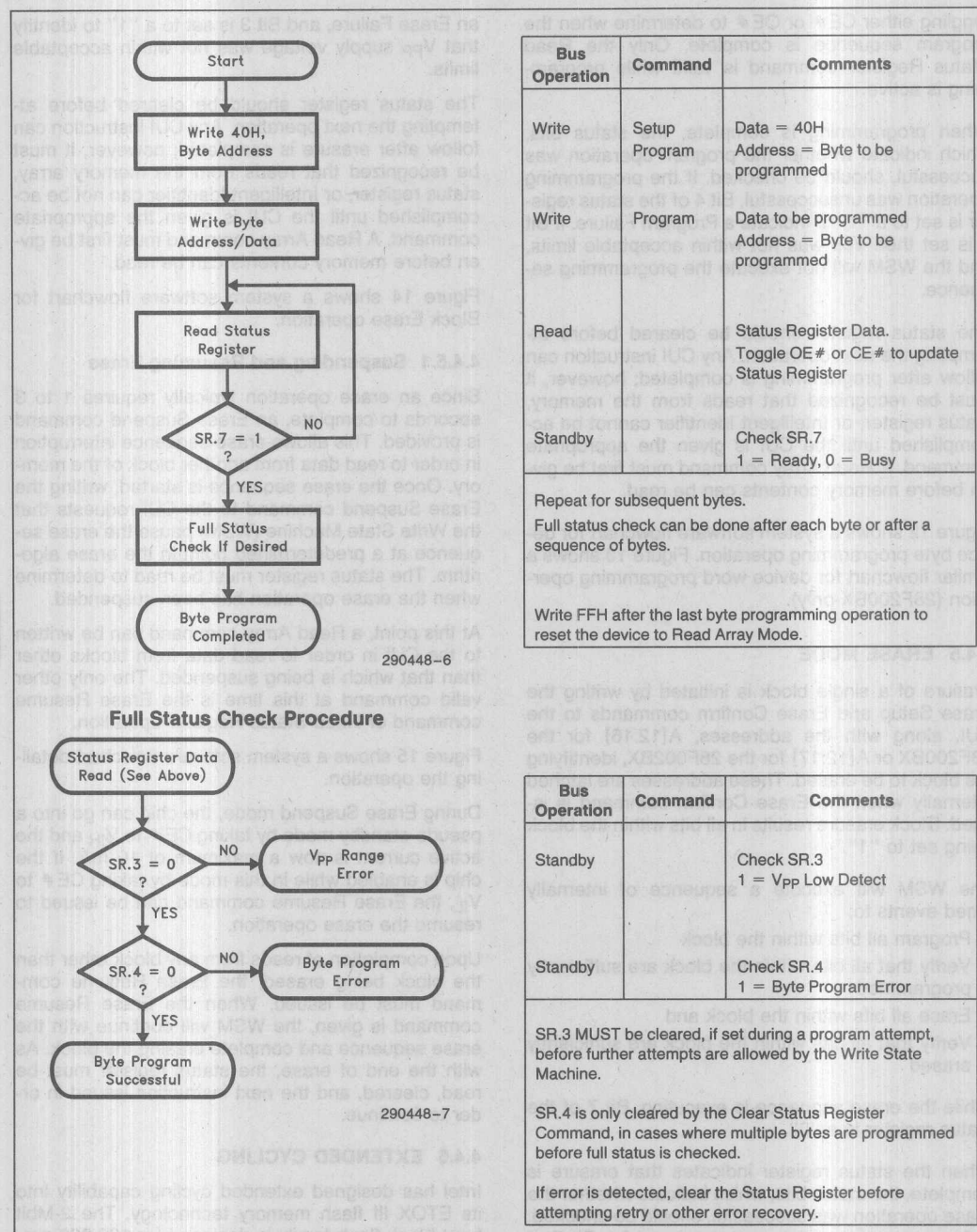


Figure 12. Automated Byte Programming Flowchart

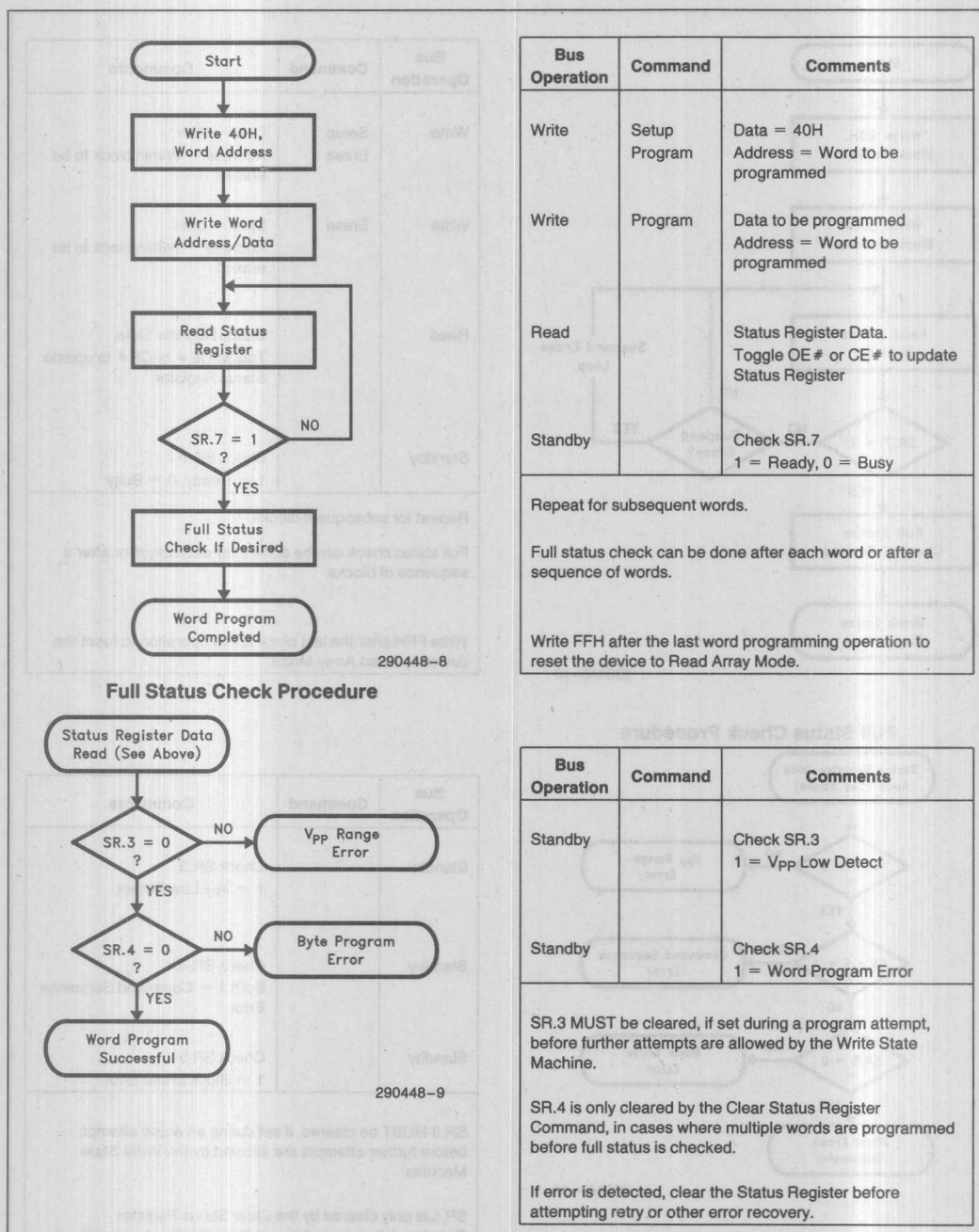
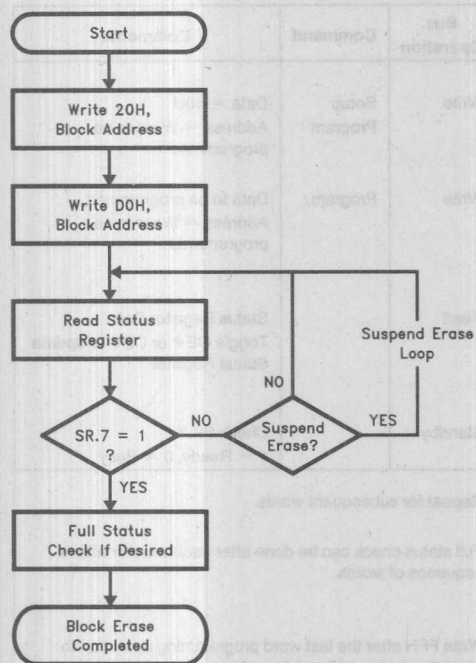
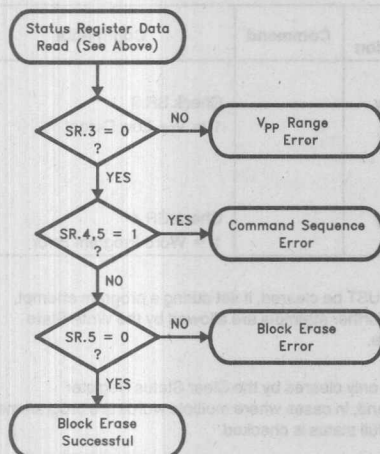


Figure 13. Automated Word Programming Flowchart



290448-10

Full Status Check Procedure



290448-11

Bus Operation	Command	Comments
Write	Setup Erase	Data = 20H Address = Within block to be erased
Write	Erase	Data = D0H Address = Within block to be erased
Read		Status Register Data. Toggle OE# or CE# to update Status Register
Standby		Check SR.7 1 = Ready, 0 = Busy

Repeat for subsequent blocks.

Full status check can be done after each block or after a sequence of blocks.

Write FFH after the last block erase operation to reset the device to Read Array Mode.

Bus Operation	Command	Comments
Standby		Check SR.3 1 = Vpp Low Detect
Standby		Check SR.4,5 Both 1 = Command Sequence Error
Standby		Check SR.5 1 = Block Erase Error

SR.3 MUST be cleared, if set during an erase attempt, before further attempts are allowed by the Write State Machine.

SR.5 is only cleared by the Clear Status Register Command, in cases where multiple blocks are erased before full status is checked.

If error is detected, clear the Status Register before attempting retry or other error recovery.

Figure 14. Automated Block Erase Flowchart

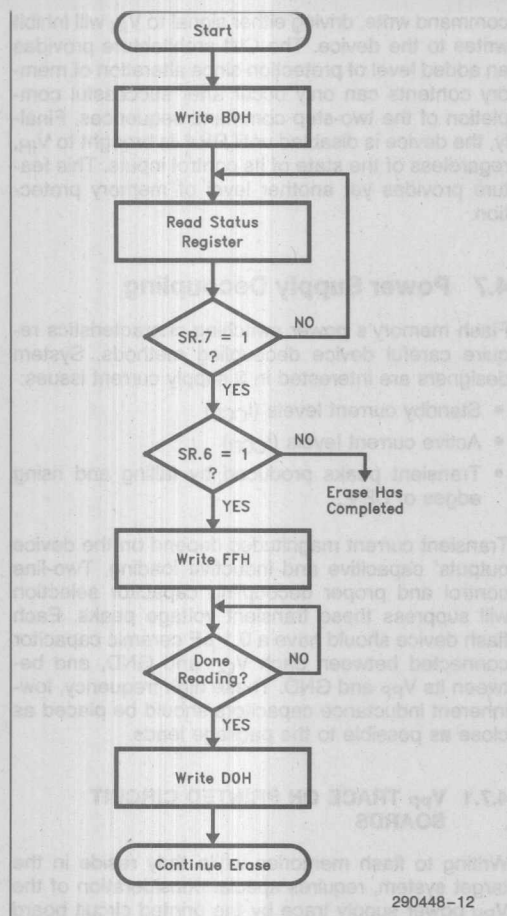


Figure 15. Erase Suspend/Resume Flowchart

4.5 Power Consumption

4.5.1 ACTIVE POWER

With CE# at a logic-low level and RP# at a logic-high level, the device is placed in the active mode. The device I_{CC} current is a maximum of 60 mA at 10 MHz with TTL input signals.

4.5.2 AUTOMATIC POWER SAVINGS

Automatic Power Savings (APS) is a low power feature during active mode of operation. The 2-Mbit family of products incorporate Power Reduction Control (PRC) circuitry which basically allows the device to put itself into a low current state when it is not being accessed. After data is read from the memory array, PRC logic controls the device's power consumption by entering the APS mode where

Bus Operation	Command	Comments
Write	Erase Suspend	Data = B0H
Read		Status Register Data. Toggle OE# or CE# to update Status Register
Standby		Check SR.7 1 = Ready
Standby		Check SR.6 1 = Suspended
Write	Read Array	Data = FFH
Read		Read array data from block other than that being erased.
Write	Erase Resume	Data = D0H

maximum I_{CC} current is 3 mA and typical I_{CC} current is 1 mA. The device stays in this static state with outputs valid until a new location is read.

4.5.3 STANDBY POWER

With CE# at a logic-high level (V_{IH}), and the CUI in read mode, the memory is placed in standby mode where the maximum I_{CC} standby current is 100 μ A with CMOS input signals. The standby operation disables much of the device's circuitry and substantially reduces device power consumption. The outputs (DQ[0:15] or DQ[0:7]) are placed in a high-impedance state independent of the status of the OE# signal. When the 2-Mbit boot block flash family is deselected during erase or program functions, the devices will continue to perform the erase or program function and consume program or erase active power until program or erase is completed.

4.5.4 RESET/DEEP POWER-DOWN

The 2-Mbit boot block flash family supports a typical I_{CC} of 0.2 μA in deep power-down mode. One of the target markets for these devices is in portable equipment where the power consumption of the machine is of prime importance. The 2-Mbit boot block flash family has a $RP\#$ pin which places the device in the deep power-down mode. When $RP\#$ is at a logic-low ($GND \pm 0.2V$), all circuits are turned off and the device typically draws 0.2 μA of V_{CC} current.

During read modes, the $RP\#$ pin going low deselects the memory and places the output drivers in a high impedance state. Recovery from the deep power-down state, requires a minimum of 400 ns to access valid data (t_{PHQV}).

During erase or program modes, $RP\#$ low will abort either erase or program operation. The contents of the memory are no longer valid as the data has been corrupted by the $RP\#$ function. As in the read mode above, all internal circuitry is turned off to achieve the 0.2 μA current level.

$RP\#$ transitions to V_{IL} or turning power off to the device will clear the status register.

This use of $RP\#$ during system reset is important with automated write/erase devices. When the system comes out of reset it expects to read from the flash memory. Automated flash memories provide status information when accessed during write/erase modes. If a CPU reset occurs with no flash memory reset, proper CPU initialization would not occur because the flash memory would be providing the status information instead of array data. Intel's Flash Memories allow proper CPU initialization following a system reset through the use of the $RP\#$ input. In this application $RP\#$ is controlled by the same RESET $\#$ signal that resets the system CPU.

4.6 Power-Up Operation

The 2-Mbit boot block flash family is designed to offer protection against accidental block erasure or programming during power transitions. Upon power-up the 2-Mbit boot block flash family is indifferent as to which power supply, V_{PP} or V_{CC} , powers-up first. Power supply sequencing is not required.

The 2-Mbit boot block flash family ensures the CUI is reset to the read mode on power-up.

In addition, on power-up the user must either drop $CE\#$ low or present a new address to ensure valid data at the outputs.

A system designer must guard against spurious writes for V_{CC} voltages above V_{LKO} when V_{PP} is

active. Since both $WE\#$ and $CE\#$ must be low for a command write, driving either signal to V_{IH} will inhibit writes to the device. The CUI architecture provides an added level of protection since alteration of memory contents can only occur after successful completion of the two-step command sequences. Finally, the device is disabled until $RP\#$ is brought to V_{IH} , regardless of the state of its control inputs. This feature provides yet another level of memory protection.

4.7 Power Supply Decoupling

Flash memory's power switching characteristics require careful device decoupling methods. System designers are interested in 3 supply current issues:

- Standby current levels (I_{CCS})
- Active current levels (I_{CCR})
- Transient peaks produced by falling and rising edges of $CE\#$.

Transient current magnitudes depend on the device outputs' capacitive and inductive loading. Two-line control and proper decoupling capacitor selection will suppress these transient voltage peaks. Each flash device should have a 0.1 μF ceramic capacitor connected between each V_{CC} and GND , and between its V_{PP} and GND . These high frequency, low-inductance capacitors should be placed as close as possible to the package leads.

4.7.1 V_{PP} TRACE ON PRINTED CIRCUIT BOARDS

Writing to flash memories while they reside in the target system, requires special consideration of the V_{PP} power supply trace by the printed circuit board designer. The V_{PP} pin supplies the flash memory cell's current for programming and erasing. One should use similar trace widths and layout considerations given to the V_{CC} power supply trace. Adequate V_{PP} supply traces and decoupling will decrease spikes and overshoots.

4.7.2 V_{CC} , V_{PP} AND $RP\#$ TRANSITIONS

The CUI latches commands as issued by system software and is not altered by V_{PP} or $CE\#$ transitions or WSM actions. Its state upon power-up, after exit from deep power-down mode or after V_{CC} transitions below V_{LKO} (Lockout voltage), is Read Array mode.

After any word/byte write or block erase operation is complete and even after V_{PP} transitions down to V_{PPL} , the CUI must be reset to Read Array mode via the Read Array command when accesses to the flash memory are desired.

ABSOLUTE MAXIMUM RATINGS*

Commercial Operating Temperature	
During Read	0°C to 70°C(1)
During Block Erase and Word/Byte Write	0°C to 70°C
Temperature Under Bias	-10°C to +80°C
Extended Operating Temperature	
During Read	-40°C to +85°C
During Block Erase and Word/Byte Write	-40°C to +85°C
Temperature Under Bias	-40°C to +85°C
Storage Temperature	
	-65°C to +125°C
Voltage on Any Pin (except V _{CC} , V _{PP} , A ₉ and RP#)	
with Respect to GND	-2.0V to +7.0V(2)
Voltage on Pin RP# or Pin A ₉ with Respect to GND	
	-2.0V to +13.5V(2, 3)
V _{PP} Program Voltage with Respect to GND during Block Erase and Word/Byte Write	
	-2.0V to +14.0V(2, 3)
V _{CC} Supply Voltage with Respect to GND	
	-2.0V to +7.0V(2)
Output Short Circuit Current	
	100 mA(4)

NOTES:

1. Operating temperature is for commercial product defined by this specification.
2. Minimum DC voltage is -0.5V on input/output pins. During transitions, this level may undershoot to -2.0V for periods <20 ns. Maximum DC voltage on input/output pins is V_{CC} + 0.5V which, during transitions, may overshoot to V_{CC} + 2.0V for periods <20 ns.
3. Maximum DC voltage on V_{PP} may overshoot to +14.0V for periods <20 ns. Maximum DC voltage on RP# or A₉ may overshoot to 13.5V for periods < 20 ns.
4. Output shorted for no more than one second. No more than one output shorted at a time.
5. 10% V_{CC} specifications reference the 28F200BX-60/28F002BX-60 in their standard test configuration, and the 28F200BX-80/28F002BX-80.
6. 5% V_{CC} specifications reference the 28F200BX-60/28F002BX-60 in their high speed test configuration.

OPERATING CONDITIONS

Symbol	Parameter	Notes	Min	Max	Units
T _A	Operating Temperature		0	70	°C
V _{CC}	V _{CC} Supply Voltage (10%)	5	4.50	5.50	V
V _{CC}	V _{CC} Supply Voltage (5%)	6	4.75	5.25	V

DC CHARACTERISTICS

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Condition
I _{LI}	Input Load Current	1			± 1.0	μA	V _{CC} = V _{CC} Max V _{IN} = V _{CC} or GND
I _{LO}	Output Leakage Current	1			± 10	μA	V _{CC} = V _{CC} Max V _{OUT} = V _{CC} or GND

NOTICE: This data sheet contains information on products in the sampling and initial production phases of development. The specifications are subject to change without notice. Verify with your local Intel Sales office that you have the latest data sheet before finalizing a design.

*WARNING: Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Condition
I _{CCS}	V _{CC} Standby Current	1, 3			1.5	mA	V _{CC} = V _{CC} Max CE# = RP# = V _{IH}
					100	μA	V _{CC} = V _{CC} Max CE# = RP# = V _{CC} ± 0.2V 28F200BX: BYTE# = V _{CC} ± 0.2V or GND
I _{CCD}	V _{CC} Deep Power-Down Current	1		0.20	1.2	μA	RP# = GND ± 0.2V
I _{CCR}	V _{CC} Read Current for 28F200BX Word-Wide and Byte-Wide Mode and 28F002BX Byte-Wide Mode	1, 5, 6, 10		20	55	mA	V _{CC} = V _{CC} Max, CE# = GND f = 10 MHz, I _{OUT} = 0 mA CMOS Inputs
				20	60	mA	V _{CC} = V _{CC} Max, CE# = V _{IL} f = 10 MHz, I _{OUT} = 0 mA TTL Inputs
I _{CCW}	V _{CC} Word Byte Write Current	1, 4			65	mA	Word Write in Progress
I _{CCE}	V _{CC} Block Erase Current	1, 4			30	mA	Block Erase in Progress
I _{CCES}	V _{CC} Erase Suspend Current	1, 2		5	10	mA	Block Erase Suspended, CE# = V _{IH}
I _{PPS}	V _{PP} Standby Current	1			± 15	μA	V _{PP} ≤ V _{CC}
I _{PPD}	V _{PP} Deep Power-Down Current	1			5.0	μA	RP# = GND ± 0.2V RP#
I _{PPR}	V _{PP} Read Current	1			200	μA	V _{PP} > V _{CC}
I _{PPW}	V _{PP} Word Write Current	1			40	mA	V _{PP} = V _{PPH} Word Write in Progress
I _{PPW}	V _{PP} Byte Write Current	1			30	mA	V _{PP} = V _{PPH} Byte Write in Progress
I _{PPE}	V _{PP} Block Erase Current	1			30	mA	V _{PP} = V _{PPH} Block Erase in Progress
I _{PPES}	V _{PP} Erase Suspend Current	1			200	μA	V _{PP} = V _{PPH} Block Erase Suspended
I _{RP#}	RP# Boot Block Unlock Current	1, 4			500	μA	RP# = V _{HH}
I _{ID}	A ₉ Intelligent Identifier Current	1, 4			500	μA	A ₉ = V _{ID}
V _{ID}	A ₉ Intelligent Identifier Voltage		11.5		13.0	V	
V _{IL}	Input Low Voltage		-0.5		0.8	V	
V _{IH}	Input High Voltage		2.0		V _{CC} + 0.5	V	
V _{OL}	Output Low Voltage				0.45	V	V _{CC} = V _{CC} Min I _{OL} = 5.8 mA

DC CHARACTERISTICS (Continued)

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Condition
V _{OH}	Output High Voltage		2.4			V	V _{CC} = V _{CC} Min I _{OH} = -2.5 mA
V _{PPL}	V _{PP} during Normal Operations	3	0.0		6.5	V	
V _{PPH}	V _{PP} during Erase/Write Operations	7	11.4	12.0	12.6	V	
V _{PPH}	V _{PP} during Erase/Write Operations	8	10.8	12.0	13.2	V	
V _{LKO}	V _{CC} Erase/Write Lock Voltage		2.0			V	
V _{HH}	RP# Unlock Voltage		11.5		13.0	V	Boot Block Write/Erase

EXTENDED TEMPERATURE OPERATING CONDITIONS

Symbol	Parameter	Notes	Min	Max	Units
T _A	Operating Temperature		-40	+85	°C
V _{CC}	V _{CC} Supply Voltage (10%)	5	4.50	5.50	V

DC CHARACTERISTICS: EXTENDED TEMPERATURE OPERATION

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Condition
I _{LI}	Input Load Current	1			±1.0	μA	V _{CC} = V _{CC} Max V _{IN} = V _{CC} or GND
I _{LO}	Output Leakage Current	1			±10	μA	V _{CC} = V _{CC} Max V _{OUT} = V _{CC} or GND
I _{CCS}	V _{CC} Standby Current	1, 3			1.5	mA	V _{CC} = V _{CC} Max CE# = RP# = V _{IH}
					100	μA	V _{CC} = V _{CC} Max CE# = RP# = V _{CC} ±0.2V 28F200BX: BYTE# = V _{CC} ±0.2V or GND
I _{CCD}	V _{CC} Deep Power-Down Current	1		0.20	8	μA	RP# = GND ±0.2V
I _{CCR}	V _{CC} Read Current for 28F200BX Word-Wide and Byte-Wide Mode and 28F002BX Byte-Wide Mode	1, 5, 6			60	mA	V _{CC} = V _{CC} Max, CE# = GND f = 10 MHz, I _{OUT} = 0 mA CMOS Inputs
					65	mA	V _{CC} = V _{CC} Max, CE# = V _{IL} f = 10 MHz, I _{OUT} = 0 mA TTL Inputs

DC CHARACTERISTICS: EXTENDED TEMPERATURE OPERATION (Continued)

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Condition
I _{CCW}	V _{CC} Word Byte Write Current	1			70	mA	Word Write in Progress
I _{CCE}	V _{CC} Block Erase Current	1			40	mA	Block Erase in Progress
I _{CCES}	V _{CC} Erase Suspend Current	1, 2		5	10	mA	Block Erase Suspended CE# = V _{IH}
I _{PPS}	V _{PP} Standby Current	1			± 15	μA	V _{PP} ≤ V _{CC}
I _{PPD}	V _{PP} Deep Power-Down Current	1			5.0	μA	RP# = GND ± 0.2V
I _{PPR}	V _{PP} Read Current	1			200	μA	V _{PP} > V _{CC}
I _{PPW}	V _{PP} Word Write Current	1			40	mA	V _{PP} = V _{PPH} Word Write in Progress
I _{PPW}	V _{PP} Byte Write Current	1			30	mA	V _{PP} = V _{PPH} Byte Write in Progress
I _{PPE}	V _{PP} Block Erase Current	1			30	mA	V _{PP} = V _{PPH} Block Erase in Progress
I _{PPES}	V _{PP} Erase Suspend Current	1			200	μA	V _{PP} = V _{PPH} Block Erase Suspended
I _{RP#}	RP# Boot Block Unlock Current	1, 4			500	μA	RP# = V _{HH}
I _{ID}	A ₉ Intelligent Identifier Current	1			500	μA	A ₉ = V _{ID}
V _{ID}	A ₉ Intelligent Identifier Voltage		11.5		13.0	V	
V _{IL}	Input Low Voltage		-0.5		0.8	V	
V _{IH}	Input High Voltage		2.0		V _{CC} + 0.5	V	
V _{OL}	Output Low Voltage				0.45	V	V _{CC} = V _{CC} Min I _{OL} = 5.8 mA
V _{OH}	Output High Voltage		2.4			V	V _{CC} = V _{CC} Min I _{OH} = -2.5 mA
V _{PPL}	V _{PP} during Normal Operations	3	0.0		6.5	V	
V _{PPH}	V _{PP} during Erase/Write Operations	7	11.4	12.0	12.6	V	
V _{PPH}	V _{PP} during Erase/Write Operations	8	10.8	12.0	13.2	V	
V _{LKO}	V _{CC} Erase/Write Lock Voltage		2.0			V	
V _{HH}	RP# Unlock Voltage		11.5		13.0	V	Boot Block Write/Erase

CAPACITANCE(4, 9) $T_A = 25^\circ\text{C}$, $f = 1\text{ MHz}$

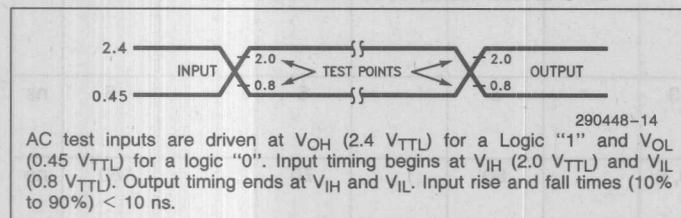
Symbol	Parameter	Typ	Max	Unit	Condition
C_{IN}	Input Capacitance	6	8	pF	$V_{IN} = 0\text{V}$
C_{OUT}	Output Capacitance	10	12	pF	$V_{OUT} = 0\text{V}$

NOTES:

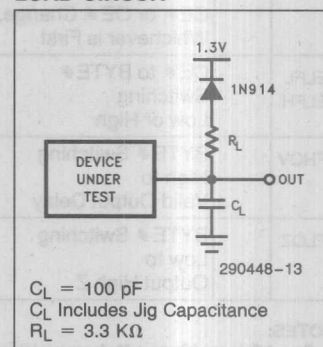
1. All currents are in RMS unless otherwise noted. Typical values at $V_{CC} = 5.0\text{V}$, $V_{PP} = 12.0\text{V}$, $T = 25^\circ\text{C}$. These currents are valid for all product versions (packages and speeds).
2. I_{CCES} is specified with the device deselected. If the device is read while in Erase Suspend Mode, current draw is the sum of I_{CCES} and I_{CCR} .
3. Block Erases and Word/Byte Writes are inhibited when $V_{PP} = V_{PPL}$ and not guaranteed in the range between V_{PPH} and V_{PPL} .
4. Sampled, not 100% tested.
5. Automatic Power Savings (APS) reduces I_{CCR} to less than 1 mA typical in static operation.
6. CMOS Inputs are either $V_{CC} \pm 0.2\text{V}$ or $\text{GND} \pm 0.2\text{V}$. TTL Inputs are either V_{IL} or V_{IH} .
7. $V_{PP} = 12.0\text{V} \pm 5\%$ for applications requiring 100,000 block erase cycles.
8. $V_{PP} = 12.0\text{V} \pm 10\%$ for applications requiring wider V_{PP} tolerances at 100 block erase cycles.
9. For the 28F002BX, address pin A_{10} follows the C_{OUT} capacitance numbers.
10. I_{CCR} typical is 25 mA for X16 active read current.

STANDARD TEST CONFIGURATION(1)

STANDARD AC INPUT/OUTPUT REFERENCE WAVEFORM

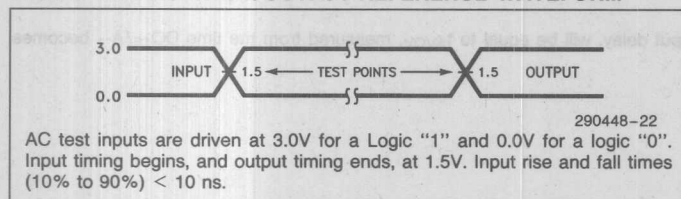


STANDARD AC TESTING LOAD CIRCUIT

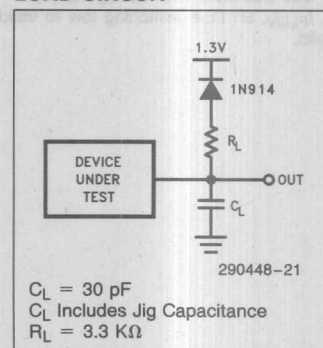


HIGH SPEED TEST CONFIGURATION(2)

HIGH SPEED AC INPUT/OUTPUT REFERENCE WAVEFORM



HIGH SPEED AC TESTING LOAD CIRCUIT



NOTES:

1. Testing characteristics for 28F200BX-60/28F002BX-60 in standard test configuration and 28F200BX-80/28F002BX-80.
2. Testing characteristics for 28F200BX-60/28F002BX-60 in high speed test configuration.

AC CHARACTERISTICS—Read Only Operations⁽¹⁾

Versions				V _{CC} ± 5%		V _{CC} ± 10%				Unit
				28F200BX-60(4) 28F002BX-60(4)		28F200BX-60(5) 28F002BX-60(5)		28F200BX-80(5) 28F002BX-80(5)		
Symbol		Parameter	Notes	Min	Max	Min	Max	Min	Max	
t _{AVAV}	t _{RC}	Read Cycle Time		60		70		80		ns
t _{AVQV}	t _{ACC}	Address to Output Delay			60		70		80	ns
t _{ELQV}	t _{CE}	CE # to Output Delay	2		60		70		80	ns
t _{PHQV}	t _{PWH}	RP # High to Output Delay			300		300		300	ns
t _{GLQV}	t _{OE}	OE # to Output Delay	2		30		35		40	ns
t _{ELQX}	t _{LZ}	CE # to Output Low Z	3	0		0		0		ns
t _{EHQZ}	t _{HZ}	CE # High to Output High Z	3		20		25		30	ns
t _{GLQX}	t _{OLZ}	OE # to Output Low Z	3	0		0		0		ns
t _{GHQZ}	t _{DF}	OE # High to Output High Z	3		20		25		30	ns
	t _{OH}	Output Hold from Addresses, CE # or OE # Change, Whichever is First	3	0		0		0		ns
t _{ELFL} t _{ELFH}		CE # to BYTE # Switching Low or High	3		5		5		5	ns
t _{FHQV}		BYTE # Switching High to Valid Output Delay	3, 6		60		70		80	ns
t _{FLQZ}		BYTE # Switching Low to Output High Z	3		20		25		30	ns

NOTES:

1. See AC Input/Output Reference Waveform for timing measurements.
2. OE # may be delayed up to t_{CE} - t_{OE} after the falling edge of CE # without impact on t_{CE}.
3. Sampled, not 100% tested.
4. See High Speed Test Configuration.
5. See Standard Test Configuration.
6. t_{FLQV}, BYTE # switching low to valid output delay, will be equal to t_{AVQV}, measured from the time DQ_{15/A-1} becomes valid.

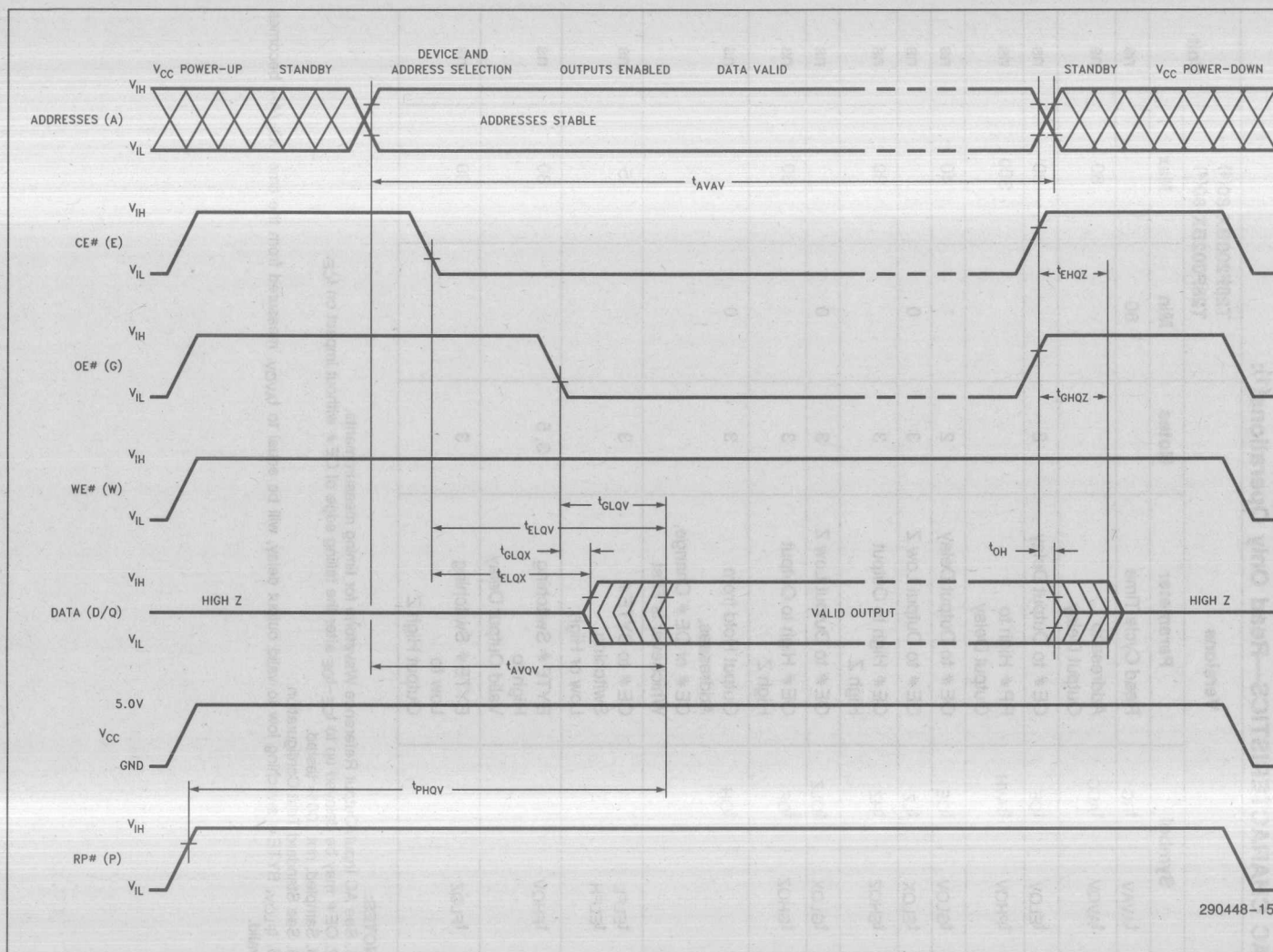
EXTENDED TEMPERATURE OPERATIONS AC CHARACTERISTICS—Read Only Operations⁽¹⁾:

Versions			T28F200BX-80(4) T28F002BX-80(4)		Unit
Symbol	Parameter	Notes	Min	Max	
t _{AVAV}	t _{RC}	Read Cycle Time	80		ns
t _{AVQV}	t _{ACC}	Address to Output Delay		80	ns
t _{ELQV}	t _{CE}	CE # to Output Delay	2	80	ns
t _{PHQV}	t _{PWH}	RP # High to Output Delay		300	ns
t _{GLQV}	t _{OE}	OE # to Output Delay	2	40	ns
t _{ELQX}	t _{LZ}	CE # to Output Low Z	3	0	ns
t _{EHQZ}	t _{HZ}	CE # High to Output High Z	3	30	ns
t _{GLQX}	t _{OLZ}	OE # to Output Low Z	3	0	ns
t _{GHQZ}	t _{DF}	OE # High to Output High Z	3	30	ns
	t _{OH}	Output Hold from Addresses, CE # or OE # Change, Whichever is First	3	0	ns
t _{ELFL} t _{ELFH}		CE # to BYTE # Switching Low or High	3	5	ns
t _{FHQV}		BYTE # Switching High to Valid Output Delay	3, 5	80	ns
t _{FLQZ}		BYTE # Switching Low to Output High Z	3	30	ns

NOTES:

1. See AC Input/Output Reference Waveform for timing measurements.
2. OE # may be delayed up to t_{CE}-t_{OE} after the falling edge of CE # without impact on t_{CE}.
3. Sampled, not 100% tested.
4. See Standard Test Configuration.
5. t_{FLQV}, BYTE # switching low to valid output delay, will be equal to t_{AVQV}, measured from the time DQ₅/A₋₁ becomes valid.

Figure 16. AC Waveforms for Read Operations



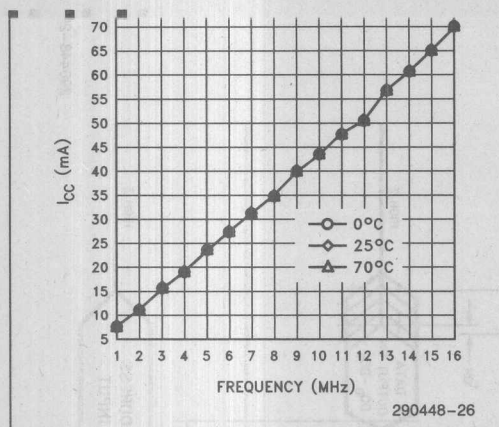


Figure 17. I_{CC} (RMS) vs Frequency (V_{CC} = 5.5V) for x16 Operation

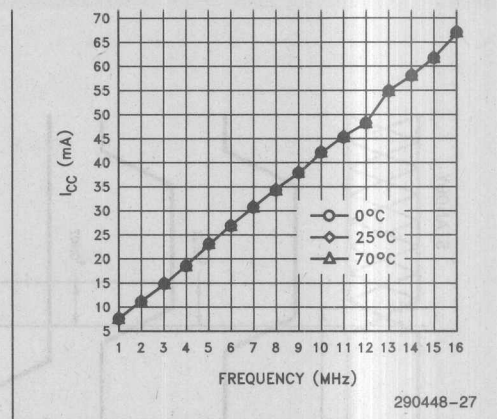


Figure 18. I_{CC} (RMS) vs Frequency (V_{CC} = 5.5V) for x8 Operation

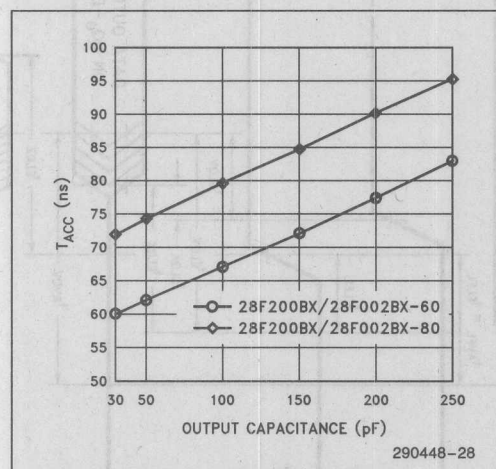
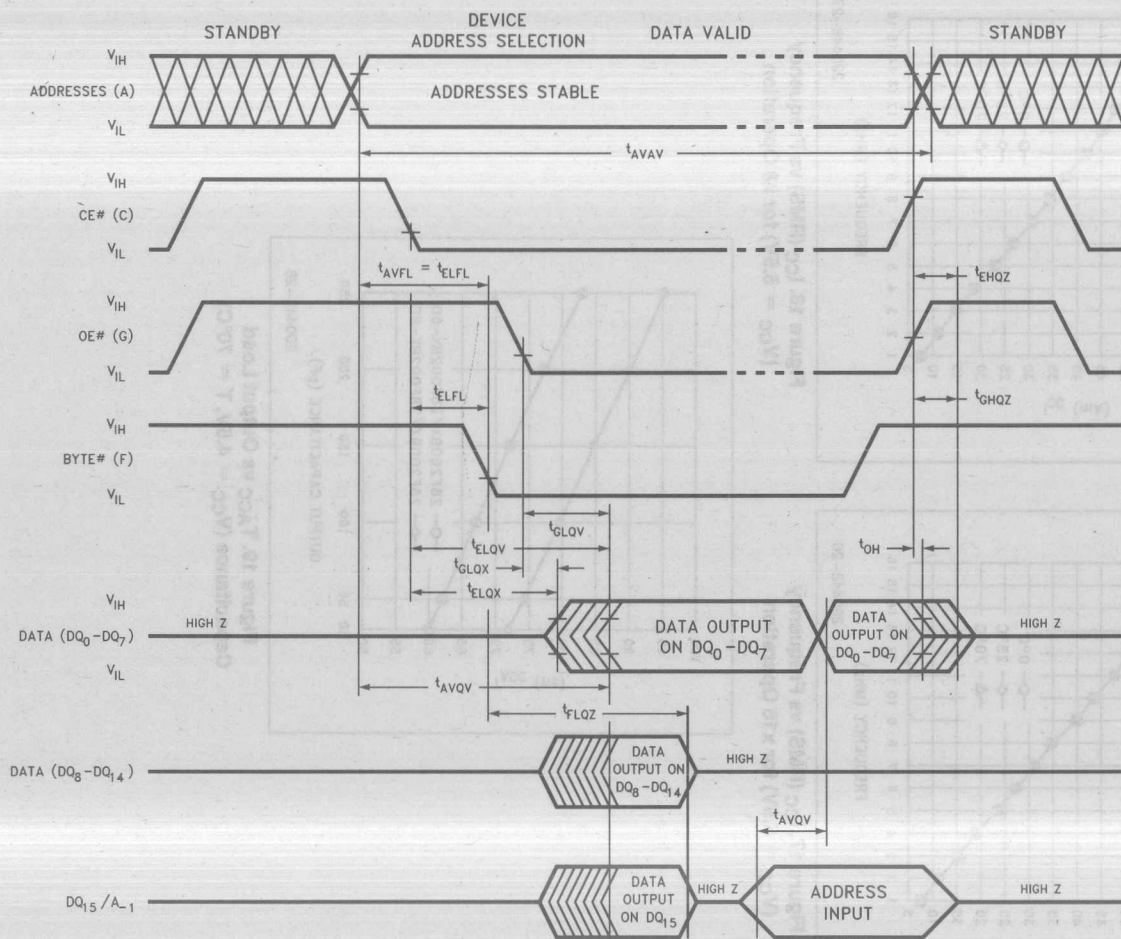


Figure 19. T_{ACC} vs Output Load Capacitance (V_{CC} = 4.5V, T = 70°C)



290448-29

Figure 20. BYTE# Timing for Both Read and Write Operations for 28F200BX

AC CHARACTERISTICS For WE #-Controlled Write Operations⁽¹⁾

Versions			V _{CC} ± 5%		V _{CC} ± 10%				Unit	
			28F200BX-60 ⁽⁹⁾ 28F002BX-60 ⁽⁹⁾		28F200BX-60 ⁽¹⁰⁾ 28F002BX-60 ⁽¹⁰⁾		28F200BX-80 ⁽¹⁰⁾ 28F002BX-80 ⁽¹⁰⁾			
Symbol		Parameter	Notes	Min	Max	Min	Max	Min	Max	
t _{AVAV}	t _{WC}	Write Cycle Time		60		70		80		ns
t _{PHWL}	t _{PS}	RP # High Recovery to WE # Going Low		215		215		215		ns
t _{ELWL}	t _{CS}	CE # Setup to WE # Going Low		0		0		0		ns
t _{PHHWH}	t _{PHS}	RP # V _{HH} Setup to WE # Going High	6, 8	100		100		100		ns
t _{VPWH}	t _{VPS}	V _{PP} Setup to WE # Going High	5, 8	100		100		100		ns
t _{AVWH}	t _{AS}	Address Setup to WE # Going High	3	50		50		50		ns
t _{DVWH}	t _{DS}	Data Setup to WE # Going High	4	60		60		60		ns
t _{WLWH}	t _{WP}	WE # Pulse Width		50		50		50		ns
t _{WHDX}	t _{DH}	Data Hold from WE # High	4	0		0		0		ns
t _{WHAX}	t _{AH}	Address Hold from WE # High	3	10		10		10		ns
t _{WHEH}	t _{CH}	CE # Hold from WE # High		10		10		10		ns
t _{WHWL}	t _{WPH}	WE # Pulse Width High		10		20		30		ns
t _{WHQV1}		Duration of Word/Byte Write Operation	2, 5	6		6		6		μs
t _{WHQV2}		Duration of Erase Operation (Boot)	2, 5, 6	0.3		0.3		0.3		s
t _{WHQV3}		Duration of Erase Operation (Parameter)	2, 5	0.3		0.3		0.3		s
t _{WHQV4}		Duration of Erase Operation (Main)	2, 5, 6	0.6		0.6		0.6		s
t _{QWL}	t _{VPH}	V _{PP} Hold from Valid SRD	5, 8	0		0		0		ns

AC CHARACTERISTICS For WE#-Controlled Write Operations⁽¹⁾ (Continued)

Versions			$V_{CC} \pm 5\%$		$V_{CC} \pm 10\%$				Unit
			28F200BX-60 ⁽⁹⁾ 28F002BX-60 ⁽⁹⁾		28F200BX-60 ⁽¹⁰⁾ 28F002BX-60 ⁽¹⁰⁾		28F200BX-80 ⁽¹⁰⁾ 28F002BX-80 ⁽¹⁰⁾		
Symbol	Parameter	Notes	Min	Max	Min	Max	Min	Max	
t_{QVPH} t_{PHH}	RP# V_{HH} Hold from Valid SRD	6, 8	0		0		0		ns
t_{PHBR}	Boot-Block Relock Delay	7, 8		100		100		100	ns

NOTES:

- Read timing characteristics during write and erase operations are the same as during read-only operations. Refer to AC characteristics during Read Mode.
- The on-chip WSM completely automates program/erase operations; program/erase algorithms are now controlled internally which includes verify and margining operations.
- Refer to command definition table for valid A_{IN} .
- Refer to command definition table for valid D_{IN} .
- Program/Erase durations are measured to valid SRD data (successful operation, SR.7=1).
- For Boot Block Program/Erase, RP# should be held at V_{HH} until operation completes successfully.
- Time t_{PHBR} is required for successful relocking of the Boot Block.
- Sampled but not 100% tested.
- See High Speed Test Configuration.
- See Standard Test Configuration.

BLOCK ERASE AND WORD/BYTE WRITE PERFORMANCE: $V_{PP} = 12.0V \pm 5\%$

Parameter	Notes	28F200BX-60 28F002BX-60			28F200BX-80 28F002BX-80			Unit
		Min	Typ ⁽¹⁾	Max	Min	Typ ⁽¹⁾	Max	
Boot/Parameter Block Erase Time	2		1.0	7		1.0	7	s
Main Block Erase Time	2		2.4	14		2.4	14	s
Main Block Byte Program Time	2		1.2	4.2		1.2	4.2	s
Main Block Word Program Time	2		0.6	2.1		0.6	2.1	s

NOTES:

- 25°C
- Excludes System-Level Overhead.

BLOCK ERASE AND WORD/BYTE WRITE PERFORMANCE: $V_{PP} = 12.0V \pm 10\%$

Parameter	Notes	28F200BX-60 28F002BX-60			28F200BX-80 28F002BX-80			Unit
		Min	Typ ⁽¹⁾	Max	Min	Typ ⁽¹⁾	Max	
Boot/Parameter Block Erase Time	2		5.8	40		5.8	40	s
Main Block Erase Time	2		14	60		14	60	s
Main Block Byte Program Time	2		6.0	20		6.0	20	s
Main Block Word Program Time	2		3.0	10		3.0	10	s

NOTES:

- 25°C
- Excludes System-Level Overhead.

EXTENDED TEMPERATURE OPERATION AC CHARACTERISTICS For WE #-Controlled Write Operations(1):

Versions(4)			T28F200BX-80(9) T28F002BX-80(9)		Unit
Symbol	Parameter	Notes	Min	Max	
t _{AVAV}	t _{WC}	Write Cycle Time	80		ns
t _{PHWL}	t _{PS}	RP# High Recovery to WE# Going Low	220		ns
t _{ELWL}	t _{CS}	CE# Setup to WE# Going Low	0		ns
t _{PHHWH}	t _{PHS}	RP# V _{HH} Setup to WE# Going High	6, 8	100	ns
t _{VPWH}	t _{VPS}	V _{PP} Setup to WE# Going High	5, 8	100	ns
t _{AVWH}	t _{AS}	Address Setup to WE# Going High	3	60	ns
t _{DVWH}	t _{DS}	Data Setup to WE# Going High	4	60	ns
t _{WLWH}	t _{WP}	WE# Pulse Width		60	ns
t _{WHDX}	t _{DH}	Data Hold from WE# High	4	0	ns
t _{WHAX}	t _{AH}	Address Hold from WE# High	3	10	ns
t _{WHEH}	t _{CH}	CE# Hold from WE# High		10	ns
t _{WHWL}	t _{WPH}	WE# Pulse Width High		20	ns
t _{WHQV1}		Duration of Word/Byte Write Operation	2, 5	7	μs
t _{WHQV2}		Duration of Erase Operation (Boot)	2, 5, 6	0.4	s
t _{WHQV3}		Duration of Erase Operation (Parameter)	2, 5	0.4	s
t _{WHQV4}		Duration of Erase Operation (Main)	2, 5, 6	0.7	s
t _{QWL}	t _{VPH}	V _{PP} Hold from Valid SRD	5, 8	0	ns
t _{QVPH}	t _{PHH}	RP# V _{HH} Hold from Valid SRD	6, 8	0	ns
t _{PHBR}		Boot-Block Relock Delay	7, 8	100	ns

NOTES:

1. Read timing characteristics during write and erase operations are the same as during read-only operations. Refer to AC characteristics during Read Mode.
2. The on-chip WSM completely automates program/erase operations; program/erase algorithms are now controlled internally which includes verify and margining operations.
3. Refer to command definition table for valid A_{IN}.
4. Refer to command definition table for valid D_{IN}.
5. Program/Erase durations are measured to valid SRD data (successful operation, SR.7 = 1).
6. For Boot Block Program/Erase, RP# should be held at V_{HH} until operation completes successfully.
7. Time t_{PHBR} is required for successful relocking of the Boot Block.
8. Sampled but not 100% tested.
9. See Standard Test Configuration.



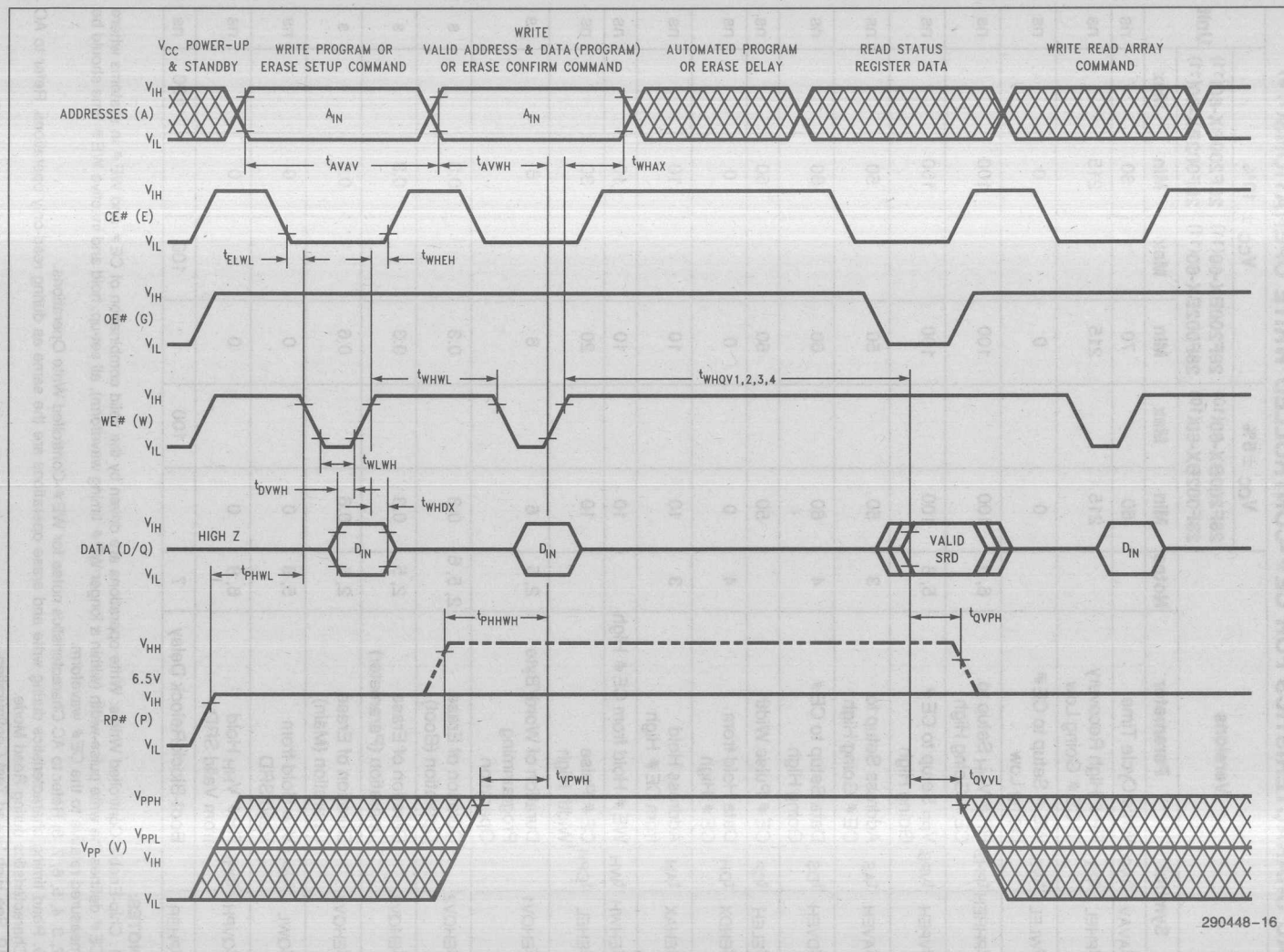
EXTENDED TEMPERATURE OPERATION
BLOCK ERASE AND WORD/BYTE WRITE PERFORMANCE: V_{pp} = 12.0V ± 5%

Parameter	Notes	T28F200BX-80 T28F002BX-80			Unit
		Min	Typ ⁽¹⁾	Max	
Boot/Parameter Block Erase Time	2		1.5	10.5	s
Main Block Erase Time	2		3.0	18	s
Main Block Byte Program Time	2		1.4	5.0	s
Main Block Word Program Time	2		0.7	2.5	s

- NOTES:
1. 25°C, 12.0V V_{pp}.
2. Excludes System-Level Overhead.

NOTES:
1. Read timing characteristics during write and erase operations are the same as during read-only operations. Refer to AC characteristics for Read Mode.
2. The on-chip write controller automatically programs/erases operations; programming algorithms are not described here, only which registers verify and monitoring operations.
3. Refer to comments' definition table for valid V_{pp}.
4. Refer to comments' definition table for valid DQ.
5. Program/erase iterations are measured to valid WRD data (successful operation only).
6. For Boot/Block/Parameter/Block, RP+ should be held at V_{pp} until operation completes successfully.
7. Time taken is related for successful locking of the Boot Block.
8. Sampled out for 100% yield.
9. See Standard Test Configuration.

Figure 21. AC Waveforms for Write and Erase Operations (WE#-Controlled Writes)



290448-16

Versions				V _{CC} ± 5%		V _{CC} ± 10%				Unit
				28F200BX-60 ⁽¹⁰⁾ 28F002BX-60 ⁽¹⁰⁾		28F200BX-60 ⁽¹¹⁾ 28F002BX-60 ⁽¹¹⁾		28F200BX-80 ⁽¹¹⁾ 28F002BX-80 ⁽¹¹⁾		
Symbol		Parameter	Notes	Min	Max	Min	Max	Min	Max	
t _{AVAV}	t _{WC}	Write Cycle Time		60		70		80		ns
t _{PHL}	t _{PS}	RP# High Recovery to CE# Going Low		215		215		215		ns
t _{WLEL}	t _{WS}	WE# Setup to CE# Going Low		0		0		0		ns
t _{PHHEH}	t _{PHS}	RP# V _{HH} Setup to CE# Going High	6, 8	100		100		100		ns
t _{VPEH}	t _{VPS}	V _{pp} Setup to CE# Going High	5, 8	100		100		100		ns
t _{AVEH}	t _{AS}	Address Setup to CE# Going High	3	50		50		50		ns
t _{DVEH}	t _{DS}	Data Setup to CE# Going High	4	60		60		60		ns
t _{ELEH}	t _{CP}	CE# Pulse Width		50		50		50		ns
t _{EHDX}	t _{DH}	Data Hold from CE# High	4	0		0		0		ns
t _{EHAX}	t _{AH}	Address Hold from CE# High	3	10		10		10		ns
t _{EWHH}	t _{WH}	WE# Hold from CE# High		10		10		10		ns
t _{EHEL}	t _{CPH}	CE# Pulse Width High		10		20		30		ns
t _{EHQV1}		Duration of Word/Byte Programming Operation	2, 5	6		6		6		μs
t _{EHQV2}		Duration of Erase Operation (Boot)	2, 5, 6	0.3		0.3		0.3		s
t _{EHQV3}		Duration of Erase Operation (Parameter)	2, 5	0.3		0.3		0.3		s
t _{EHQV4}		Duration of Erase Operation (Main)	2, 5	0.6		0.6		0.6		s
t _{QWL}	t _{VPH}	V _{pp} Hold from Valid SRD	5, 8	0		0		0		ns
t _{QVPH}	t _{PHH}	RP# V _{HH} Hold from Valid SRD	6, 8	0		0		0		ns
t _{PHBR}		Boot-Block Relock Delay	7		100		100		100	ns

NOTES:

1. Chip-Enable Controlled Writes: Write operations are driven by the valid combination of CE# and WE# in systems where CE# defines the write pulse-width (within a longer WE# timing waveform), all set-up, hold and inactive WE# time should be measured relative to the CE# waveform.

2, 3, 4, 5, 6, 7, 8: Refer to AC Characteristics notes for WE#-Controlled Write Operations.

9. Read timing characteristics during write and erase operations are the same as during read-only operations. Refer to AC Characteristics during Read Mode.

10. See High Speed Test Configuration.

11. See Standard Test Configuration.

EXTENDED TEMPERATURE OPERATION AC CHARACTERISTICS FOR CE#-CONTROLLED WRITE OPERATIONS(1, 9)

Versions			T28F200BX-80(10) T28F002BX-80(10)		Unit
Symbol	Parameter	Notes	Min	Max	
t _{AVAV}	t _{WC}	Write Cycle Time	80		ns
t _{PHL}	t _{PS}	RP# High Recovery to CE# Going Low	220		ns
t _{WLEL}	t _{WS}	WE# Setup to CE# Going Low	0		ns
t _{PHHEH}	t _{PHS}	RP# V _{HH} Setup to CE# Going High	100		ns
t _{VPEH}	t _{VPS}	V _{PP} Setup to CE# Going High	100		ns
t _{AVEH}	t _{AS}	Address Setup to CE# Going High	60		ns
t _{DVEH}	t _{DS}	Data Setup to CE# Going High	60		ns
t _{ELEH}	t _{CP}	CE# Pulse Width	60		ns
t _{EHDX}	t _{DH}	Data Hold from CE# High	0		ns
t _{EHAX}	t _{AH}	Address Hold from CE# High	10		ns
t _{EHWH}	t _{WH}	WE# Hold from CE# High	10		ns
t _{EH}	t _{CPH}	CE# Pulse Width High	20		ns
t _{EHQV1}		Duration of Word/Byte Programming Operation	7		μs
t _{EHQV2}		Duration of Erase Operation (Boot)	0.4		s
t _{EHQV3}		Duration of Erase Operation (Parameter)	0.4		s
t _{EHQV4}		Duration of Erase Operation (Main)	0.7		s
t _{QWL}	t _{VPH}	V _{PP} Hold from Valid SRD	0		ns
t _{QVPH}	t _{PHH}	RP# V _{HH} Hold from Valid SRD	0		ns
t _{PHBR}		Boot-Block Relock Delay	7	100	ns

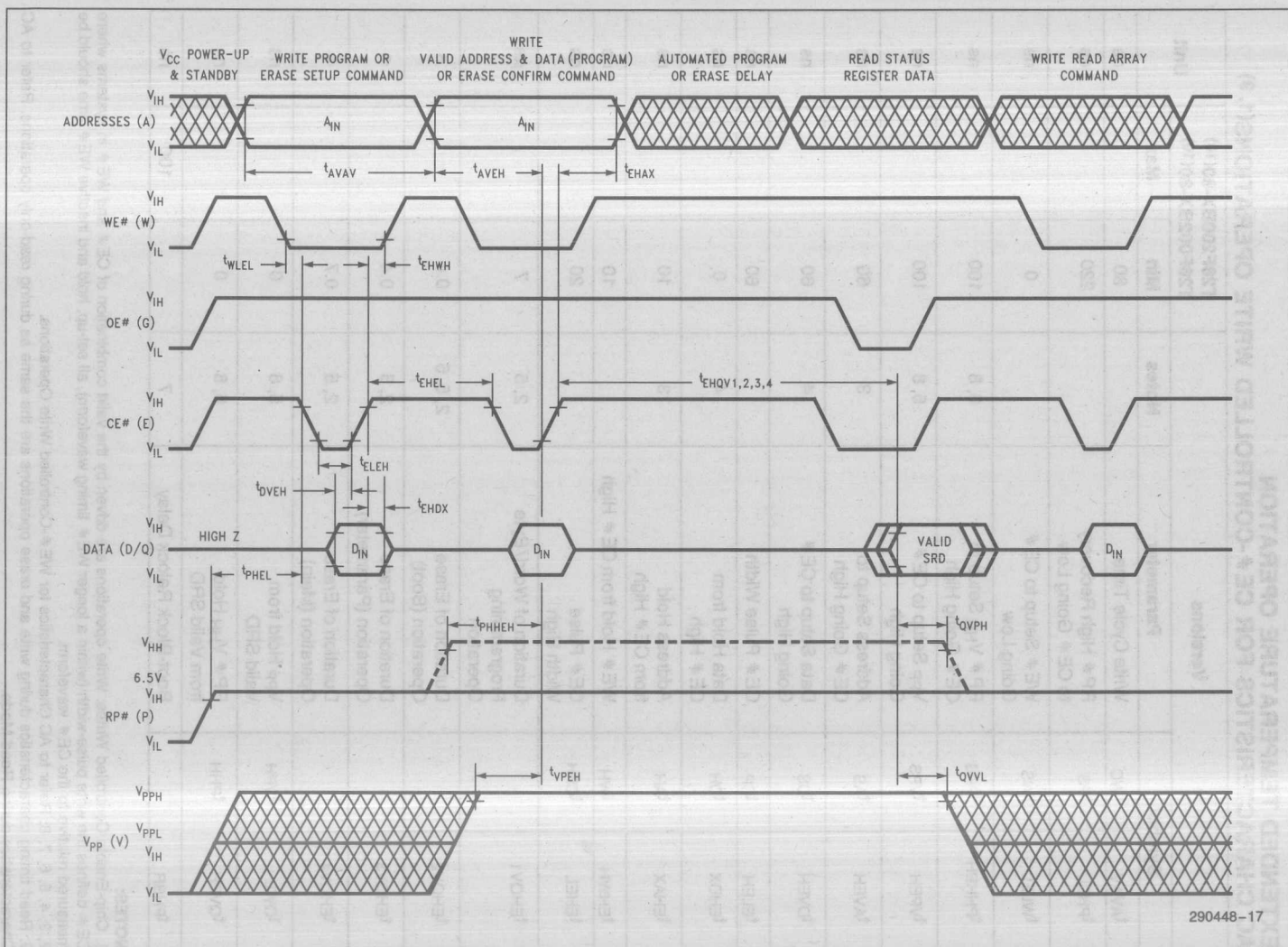
NOTES:

1. Chip-Enable Controlled Writes: Write operations are driven by the valid combination of CE# and WE# in systems where CE# defines the write pulse-width (within a longer WE# timing waveform), all set-up, hold and inactive WE# time should be measured relative to the CE# waveform.

2, 3, 4, 5, 6, 7, 8: Refer to AC Characteristics for WE#-Controlled Write Operations.

9. Read timing characteristics during write and erase operations are the same as during read-only operations. Refer to AC Characteristics during Read Mode.

10. See Standard Test Configuration.



ORDERING INFORMATION

E 2 8 F 2 0 0 B X - T 6 0											
OPERATING TEMPERATURE T = EXTENDED TEMP BLANK = COMMERCIAL TEMP				PACKAGE E = STANDARD 56 LEAD TSOP PA = 44 LEAD PSOP TB = 44 LEAD PSOP, EXTENDED TEMP				ACCESS SPEED (ns) 60 ns 80 ns			
290448-18											
Valid Combinations:											
E28F200BX-T60	PA28F200BX-T60			TE28F200BX-T80	TB28F200BX-T80						
E28F200BX-B60	PA28F200BX-B60			TE28F200BX-B80	TB28F200BX-B80						
E28F200BX-T80	PA28F200BX-T80			TE28F200BX-B80	TB28F200BX-B80						
E28F200BX-B80	PA28F200BX-B80										

	E	2	8	F	0	0	2	B	X	-	T	6	0
--	---	---	---	---	---	---	---	---	---	---	---	---	---

OPERATING TEMPERATURE

T = EXTENDED TEMP

BLANK = COMMERCIAL TEMP

PACKAGE

E = STANDARD 40 LEAD TSOP

ACCESS SPEED (ns)

60 ns

80 ns

290448-23

Valid Combinations:

E28F002BX-T60	
E28F002BX-B60	
E28F002BX-T80	TE28F002BX-T80
E28F002BX-B80	TE28F002BX-B80

4

ADDITIONAL INFORMATION

28F400BX/28F004BX Datasheet	290451
28F200BXL/28F002BXL Datasheet	290449
28F400BXL/28F004BXL Datasheet	290450
AP-363 "Extended Flash BIOS Design for Portable Computers"	292098
ER-28 "ETOX-III Flash Memory Technology"	204012
ER-29 "The Intel 2/4-Mbit Boot Block Flash Memory Family"	294013

Order Number

REVISION HISTORY

Number	Description
-002	Removed -70 speed bin Integrated -70 characteristics into -60 speed bin Added Extended Temperature characteristics Modified BYTE# Timing Diagram Improved t _{PHQV} , RP# High to Output Delay and t _{PHL} , RP# High Recovery to CE# going low specifications
-003	PWD changed to RP# for JEDEC standardization compatibility. Combined V _{CC} Read current for 28F200BX Word-wide mode and Byte-wide mode, and 28F002BX Byte-wide mode in DC Characteristics tables. Change I _{PPS} current spec from ±10 μA to ±15 μA in DC Characteristics tables. Improved I _{CCR} and I _{CCW} in DC Characteristics: Extended Temperature Operation table. Improved t _{AVAV} , t _{AVQV} , t _{ELQV} , t _{GLQV} , t _{EHQZ} , t _{GHQZ} , t _{FHQV} and t _{FLQZ} specifications for Extended Temperature Operations AC Characteristics—Read and Write Operations.

ADVANCE INFORMATION



PRELIMINARY

28F200BX-TL/BL, 28F002BX-TL/BL 2-MBIT (128K x 16, 256K x 8) LOW POWER BOOT BLOCK FLASH MEMORY FAMILY

- Low Voltage Operation for Very Low Power Portable Applications
 - $V_{CC} = 3.3V \pm 0.3V$
- x8/x16 Input/Output Architecture
 - 28F200BX-TL, 28F200BX-BL
 - For High Performance and High Integration 16-bit and 32-bit CPUs
- x8-only Input/Output Architecture
 - 28F002BX-TL, 28F002BX-BL
 - For Space Constrained 8-bit Applications
- Optimized High Density Blocked Architecture
 - One 16-KB Protected Boot Block
 - Two 8-KB Parameter Blocks
 - One 96-KB Main Block
 - One 128-KB Main Block
 - Top or Bottom Boot Locations
- Extended Cycling Capability
 - 10,000 Block Erase Cycles
- Automated Word/Byte Write and Block Erase
 - Command User Interface
 - Status Registers
 - Erase Suspend Capability
- SRAM-Compatible Write Interface
- Automatic Power Savings Feature
 - 0.8 mA Typical I_{CC} Active Current in Static Operation
- Very High-Performance Read
 - 150 ns Maximum Access Time
 - 65 ns Maximum Output Enable Time
- Low Power Consumption
 - 15 mA Typical Active Read Current
- Reset/Deep Power-Down Input
 - 0.2 μA I_{CC} Typical
 - Acts as Reset for Boot Operations
- Write Protection for Boot Block
- Hardware Data Protection Feature
 - Erase/Write Lockout during Power Transitions
- Industry Standard Surface Mount Packaging
 - 28F200BX-L: JEDEC ROM Compatible 44-Lead PSOP 56-Lead TSOP
 - 28F002BX-L: 40-Lead TSOP
- 12V Word/Byte Write and Block Erase
 - $V_{pp} = 12V \pm 5\%$ Standard
- ETOX III Flash Technology
 - 3.3V Read
- Independent Software Vendor Support

REVISION HISTORY

Number	Description
002	Removed -70 speed bin Integrated -70 characteristics into -80 speed bin Added Extended Temperature characteristics Modified BYTE Timing Diagram Increased I_{PP} High to Output Delay and Level, I_{PP} High Recovery, I_{CC} going low specifications
003	I_{PP} changed to I_{PP} for JEDEC standardization compatibility Changed V_{CC} Read current for 28F200BX Word write mode and Byte write mode and 28F002BX Byte write mode in DC Characteristics tables Changed I_{PP} current spec from $\pm 10 \mu A$ to $\pm 15 \mu A$ in DC Characteristics table Improved I_{CC} and I_{PP} in DC Characteristics Extended Temperature Operation table Improved I_{PP} Low, I_{PP} Low for Low Power mode and in DC Characteristics for Extended Temperature Operation AC Characteristics—Read and Write Operations

Intel's 2-Mbit Low Power Flash Memory Family is an extension of the Boot Block Architecture which includes block-selective erasure, automated write and erase operations and standard microprocessor interface. The 2-Mbit Flash Memory Family enhances the Boot Block Architecture by adding more density and blocks, x8/x16 input/output control, very low power, very high speed, an industry standard ROM compatible pinout and surface mount packaging. The 2-Mbit Low Power Flash Family opens a new capability for 3V battery-operated portable systems and allows for an easy upgrade to Intel's 4-Mbit Low Power Boot Block Flash Memory Family.

The Intel 28F200BX-TL/BL are 16-bit wide flash memory offerings. These high density flash memories provide user selectable bus operation for either 8-bit or 16-bit applications. The 28F200BX-TL and 28F200BX-BL are 2,097,152-bit non-volatile memories organized as either 262,144 bytes or 131,072 words of information. They are offered in 44-Lead plastic SOP and 56-Lead TSOP packages. The x8/x16 pinout conforms to the industry standard ROM/EPROM pinout.

The Intel 28F002BX-TL/BL are 8-bit wide flash memories with 2,097,152 bits organized as 262,144 bytes of information. They are offered in a 40-Lead TSOP package, which is ideal for space-constrained portable systems.

These devices use an integrated Command User Interface (CUI) and Write State Machine (WSM) for simplified word/byte write and block erasure. The 28F200BX-TL/28F002BX-TL provide block locations compatible with Intel's low voltage MCS-186 family, i386™, i486™ microprocessors. The 28F200BX-BL/28F002BX-BL provide compatibility with Intel's 80960KX and 80960SX families as well as other low voltage embedded microprocessors.

The boot block includes a data protection feature to protect the boot code in critical applications. With a maximum access time of 150 ns, these 2-Mbit flash devices are very high performance low power memories which interface to a wide range of low power microprocessors and microcontrollers. A deep power-down mode lowers the total V_{CC} power consumption to 0.66 μ W. This is critical in handheld battery powered systems such as Handy Phones. For very high speed applications using a 5V supply, refer to the Intel 28F200BX-T/B, 28F002BX-T/B 2-Mbit Boot Block Flash Memory Family datasheet.

Manufactured on Intel's 0.8 micron ETOX III process, the 2-Mbit low power flash memory family provides world class quality, reliability and cost-effectiveness at the 2-Mbit density level.

1.0 PRODUCT FAMILY OVERVIEW

Throughout this datasheet 28F200BX-L refers to both the 28F200BX-TL and 28F200BX-BL devices and 28F002BX-L refers to both the 28F002BX-TL and 28F002BX-BL devices. The 2-Mbit flash family refers to both the 28F200BX-L and 28F002BX-L products. This datasheet comprises the specifications for four separate products in the 2-Mbit flash memory family. Section 1 provides an overview of the 2-Mbit flash memory family including applications, pinouts and pin descriptions. Sections 2 and 3 describe in detail the specific memory organizations for the 28F200BX-L and 28F002BX-L products respectively. Section 4 combines a description of the family's principles of operations. Finally, section 5 describes the family's operating specifications.

PRODUCT FAMILY

x8/x16 Products	x8-Only Products
28F200BX-TL 28F200BX-BL	28F002BX-TL 28F002BX-BL

1.1 Main Features

The 28F200BX-L/28F002BX-L low power boot block flash memory family is a very low power and very high performance 2-Mbit (2,097,152 bit) memory family organized as either 128 Kwords (131,072 words) of 16 bits each or 256 Kbytes (262,144 bytes) of 8 bits each.

Five Separately Erasable Blocks including a **Hardware-Lockable boot block** (16,384 Bytes), **two parameter blocks** (8,192 Bytes each) and **two main blocks** (1 block of 98,304 Bytes and 1 block of 131,072 Bytes) are included on the 2-Mbit family. An erase operation erases one of the 5 blocks in typically 3.4 seconds and the boot or parameter blocks in typically 2.0 seconds, independent of the remaining blocks. Each block can be independently erased and programmed 10,000 times.

The Boot Block is located at either the top (28F200BX-TL, 28F002BX-TL) or the bottom (28F200BX-BL, 28F002BX-BL) of the address map in order to accommodate different microprocessor protocols for boot code location. The **hardware lockable boot block** provides the most secure code

storage. The boot block is intended to store the kernel code required for booting-up a system. When the RP# pin is between 11.4V and 12.6V the boot block is unlocked and program and erase operations can be performed. When the RP# pin is at or below 4.1V the boot block is locked and program and erase operations to the boot block are ignored.

The 28F200BX-L products are available in the ROM/EPROM compatible pinout and housed in the 44-Lead PSOP (Plastic Small Outline) package and the 56-Lead TSOP (Thin Small Outline, 1.2 mm thick) package as shown in Figures 3 and 4. The 28F002BX-L products are available in the 40-Lead TSOP (1.2 mm thick) package as shown in Figure 5.

The **Command User Interface (CUI)** serves as the interface between the microprocessor or microcontroller and the internal operation of the 28F200BX-L and 28F002BX-L flash memory products.

Program and Erase Automation allow program and erase operations to be executed using a two-write command sequence to the CUI. The internal Write State Machine (WSM) automatically executes the algorithms and timings necessary for program and erase operations, including verifications, thereby unburdening the microprocessor or microcontroller. Writing of memory data is performed in word or byte increments for the 28F200BX-L family and in byte increments for the 28F002BX-L family typically within 11 μ s.

The **Status Register (SR)** indicates the status of the WSM and whether the WSM successfully completed the desired program or erase operation.

Maximum Access Time of **150 ns (T_{ACC})** is achieved over the commercial temperature range (0°C to +70°C), over V_{CC} supply voltage range (3.0V to 3.6V, 4.5V to 5.5V) and 50 pF output load.

I_{pp} Program current is 40 mA for x16 operation and 30 mA for x8 operation. I_{pp} Erase current is 30 mA maximum. V_{pp} erase and programming voltage is 11.4V to 12.6V ($V_{pp} = 12V \pm 5\%$) under all operating conditions.

Typical I_{CC} Active Current of 15 mA is achieved for the x16 products and the x8 products.

The 2-Mbit flash family is also designed with an Automatic Power Savings (APS) feature to minimize system battery current drain and allow for extremely low power designs. Once the device is accessed to read the array data, APS mode will immediately put the memory in static mode of operation where I_{CC} active current is typically 0.8 mA until the next read is initiated.

When the CE# and RP# pins are at V_{CC} and the BYTE# pin (28F200BX-L-only) is at either V_{CC} or GND the **CMOS Standby** mode is enabled where I_{CC} is typically 40 μA .

A **Deep Power-down Mode** is enabled when the RP# pin is at ground minimizing power consumption and providing write protection during power-up conditions. I_{CC} current during deep power-down mode is 0.20 μA typical. An initial maximum access time or Reset Time of 700 ns is required from RP# switching until outputs are valid. Equivalently, the device has a maximum wake-up time of 580 ns until writes to the Command User Interface are recognized. When RP# is at ground the WSM is reset, the Status Register is cleared and the entire device is protected from being written to. This feature prevents data corruption and protects the code stored in the device during system reset. The system Reset pin can be tied to RP# to reset the memory to normal read mode upon activation of the Reset pin. When the CPU enters reset mode, it expects to read the contents of a memory location. Furthermore, with on-chip program/erase automation in the 2-Mbit family and the RP# functionality for data protection, after the CPU is reset and even if a program or erase command is issued, the device will not recognize any operation until RP# returns to its normal state.

For the 28F200BX-L, Byte-wide or Word-wide Input/Output Control is possible by controlling the BYTE# pin. When the BYTE# pin is at a logic low the device is in the byte-wide mode (x8) and data is read and written through DQ[0:7]. During the byte-wide mode, DQ[8:14] are tri-stated and DQ₁₅/A₋₁ becomes the lowest order address pin. When the BYTE# pin is at a logic high the device is in the word-wide mode (x16) and data is read and written through DQ[0:15].

1.2 Applications

The 2-Mbit low power boot block flash memory family combines high density, 3V operation, high performance, cost-effective flash memories with blocking and hardware protection capabilities. Its flexibility and versatility will reduce costs throughout the product life cycle. Flash memory is ideal for Just-In-Time

production flow, reducing system inventory and costs, and eliminating component handling during the production phase. During the product life cycle, when code updates or feature enhancements become necessary, flash memory will reduce the update costs by allowing either a user-performed code change via floppy disk or a remote code change via a serial link. The 2-Mbit boot block flash memory family provides full function, blocked flash memories suitable for a wide range of applications. These applications include **Extended PC BIOS**, **Handy Digital Cellular Phone** program and data storage and various other portable embedded applications where both program and data storage are required.

Reprogrammable systems such as Notebook and Palmtop computers, are ideal applications for the 2-Mbit low power flash products. Portable and handheld personal computer applications are becoming more complex with the addition of power management software to take advantage of the latest microprocessor technology, the availability of ROM-based application software, pen tablet code for electronic handwriting, and diagnostic code. Figure 1 shows an example of a 28F200BX-TL application.

This increase in software sophistication augments the probability that a code update will be required after the PC is shipped. The 2-Mbit low power flash memory products provide an inexpensive update solution for the notebook and handheld personal computers while extending their product lifetime. Furthermore, the 2-Mbit flash memory products' deep power-down mode provides added flexibility for these battery-operated portable designs which require operation at extremely low power levels.

The 2-Mbit low power flash products also provide excellent design solutions for Handy Digital Cellular Phone applications requiring high density storage, high performance capabilities coupled with low voltage operation, and a small form factor package (x8-only bus). The 2-Mbit's blocking scheme allows for an easy segmentation of the embedded code with: 16 Kbytes of Hardware-Protected Boot code, 2 Main Blocks of program code and 2 Parameter Blocks of 8 Kbytes each for frequently updatable data storage and diagnostic messages (e.g., phone numbers, authorization codes). Figure 2 is an example of such an application with the 28F002BX-TL.

These are a few actual examples of the wide range of applications for the 2-Mbit Low Power Boot Block flash memory family which enables system designers to achieve the best possible product design. Only your imagination limits the applicability of such a versatile low power product family.

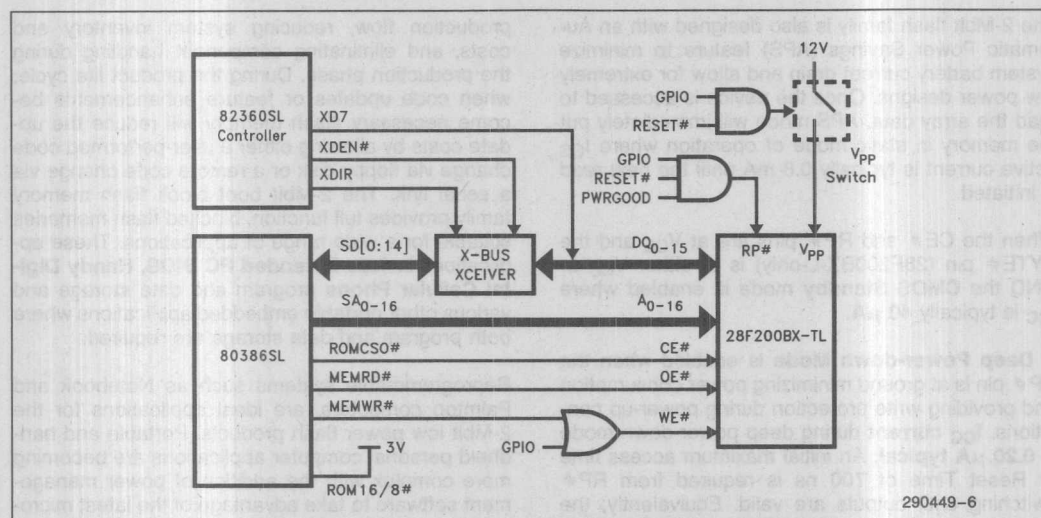


Figure 1. 28F200BX-TL Interface to INTEL386SL 3.3V Microprocessor Superset

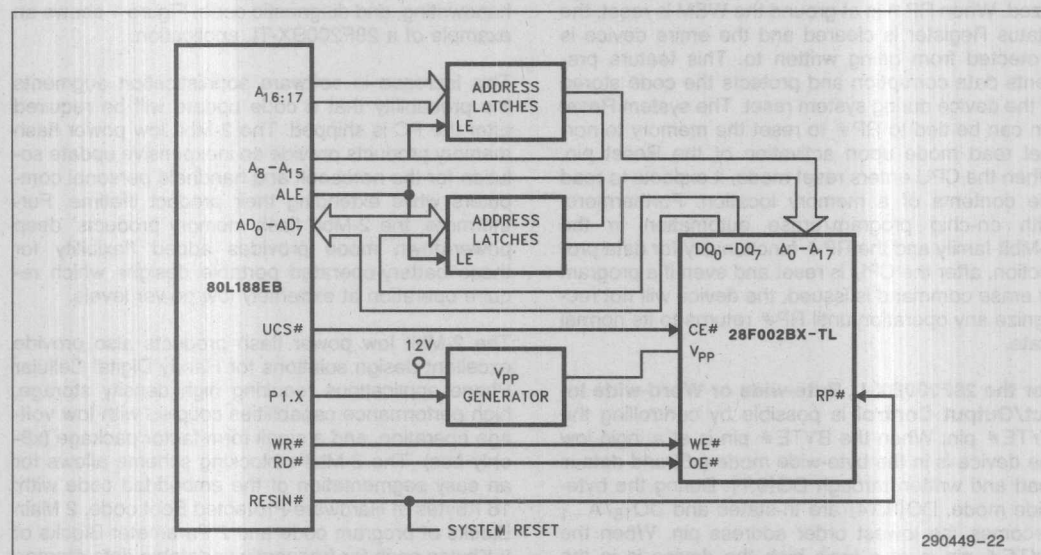


Figure 2. 28F002BX-TL Interface to INTEL 80L188EB, Low Voltage 8-Bit Embedded Microprocessor

1.3 PINOUTS

The 28F200BX-L 44-Lead PSOP pinout follows the industry standard ROM/EPROM pinout as shown in Figure 3 with an upgrade to the 28F400BX-L (4-Mbit low power flash family). Furthermore, the 28F200BX-L 56-Lead TSOP pinout shown in

Figure 4 provides density upgrade to the 28F400BX-L and to future higher density boot block memories.

The 28F002BX-L 40-Lead TSOP pinout shown in Figure 5 is 100% compatible and has a density upgrade to the 28F004BX-L 4-Mbit Low Power Boot Block flash memory.

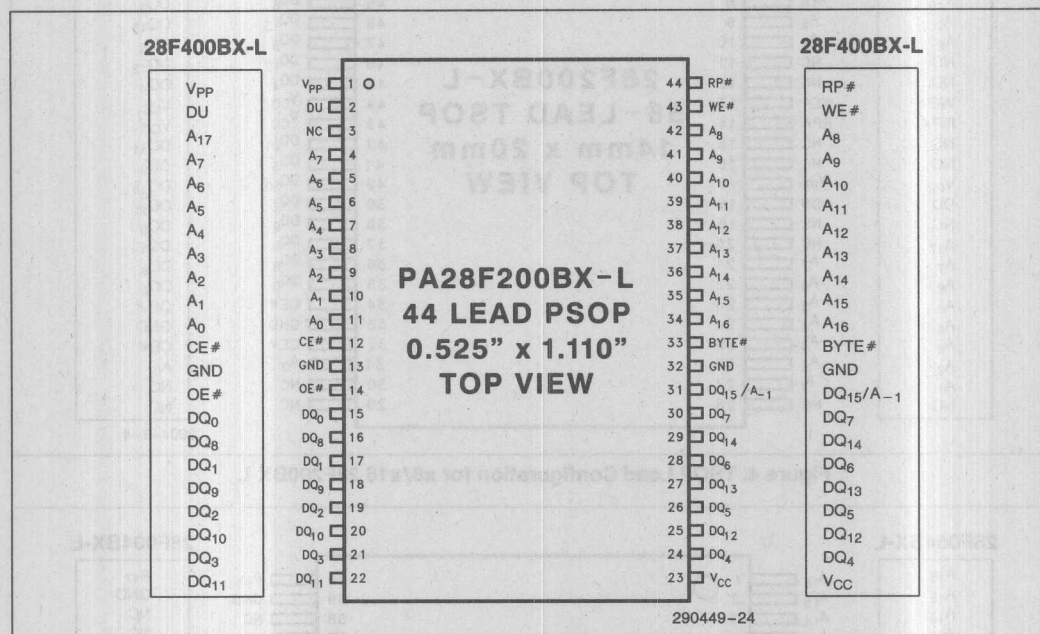


Figure 3. PSOP Lead Configuration for x8/x16 28F200BX-L

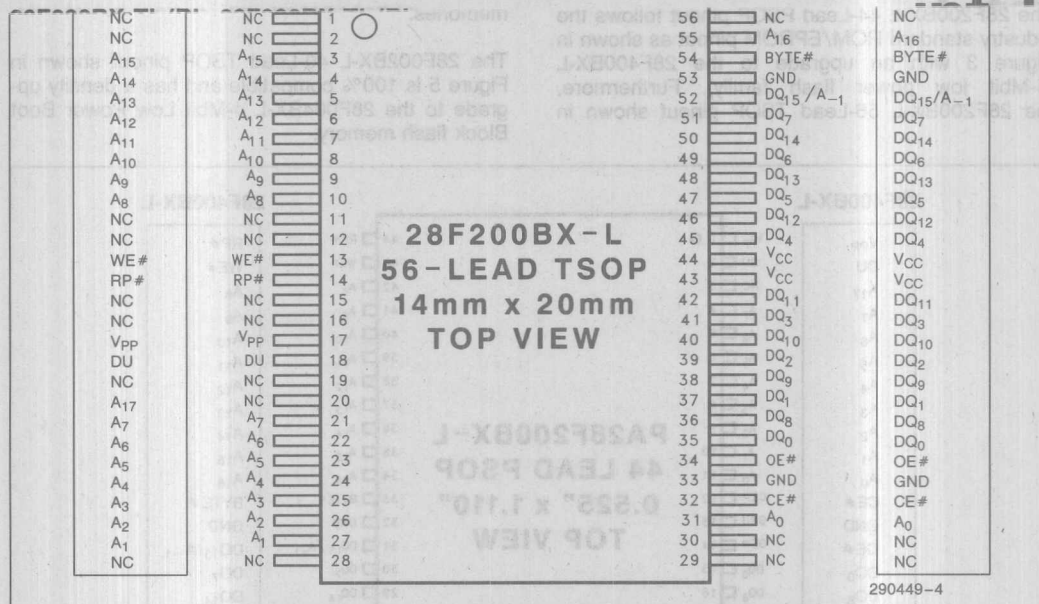


Figure 4. TSOP Lead Configuration for x8/x16 28F200BX-L

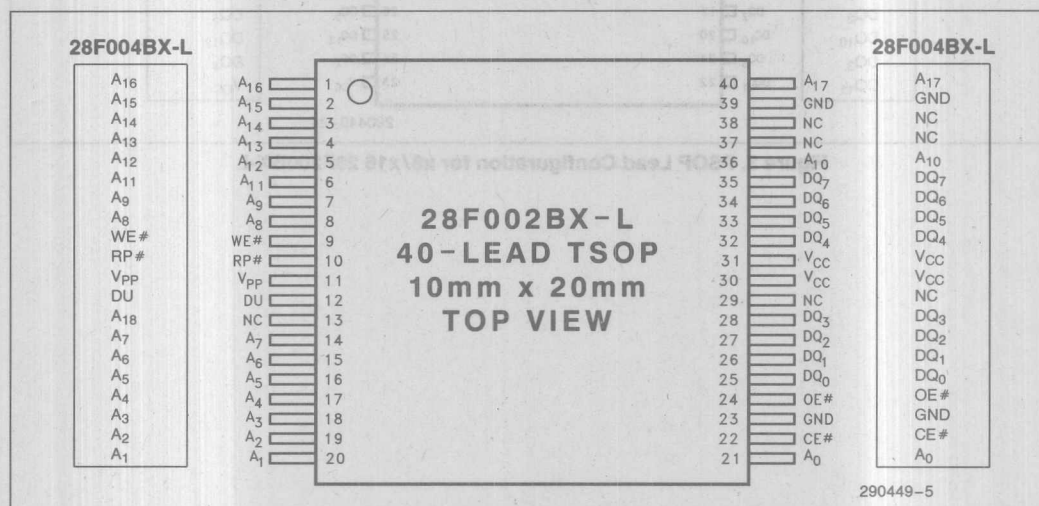


Figure 5. TSOP Lead Configuration for x8 28F002BX-L

1.4 Pin Descriptions for x8/x16 28F200BX-L

Symbol	Type	Name and Function
A ₀ –A ₁₆	I	ADDRESS INPUTS for memory addresses. Addresses are internally latched during a write cycle.
A ₉	I	ADDRESS INPUT: When A ₉ is at 12V the signature mode is accessed. During this mode A ₀ decodes between the manufacturer and device ID's. When BYTE# is at a logic low only the lower byte of the signatures are read. DQ ₁₅ /A _{–1} is a don't care in the signature mode when BYTE# is low.
DQ ₀ –DQ ₇	I/O	DATA INPUTS/OUTPUTS: Inputs array data on the second CE# and WE# cycle during a program command. Inputs commands to the command user interface when CE# and WE# are active. Data is internally latched during the write and program cycles. Outputs array, Intelligent Identifier and Status Register data. The data pins float to tri-state when the chip is deselected or the outputs are disabled.
DQ ₈ –DQ ₁₅	I/O	DATA INPUT/OUTPUTS: Inputs array data on the second CE# and WE# cycle during a program command. Data is internally latched during the write and program cycles. Outputs array data. The data pins float to tri-state when the chip is deselected or the outputs are disabled as in the byte-wide mode (BYTE# = "0"). In the byte-wide mode DQ ₁₅ /A _{–1} becomes the lowest order address for data output on DQ ₀ –DQ ₇ .
CE#	I	CHIP ENABLE: Activates the device's control logic, input buffers, decoders and sense amplifiers. CE# is active low; CE# high deselects the memory device and reduces power consumption to standby levels. If CE# and RP# are high, but not at a CMOS high level, the standby current will increase due to current flow through the CE# and RP# input stages.
RP#	I	RESET/DEEP POWER-DOWN: Provides Three-State control. Puts the device in deep power-down mode. Locks the boot block from program/erase. When RP# is at logic high level and equals 4.1V maximum the boot block is locked and cannot be programmed or erased. When RP# = 11.4V minimum the boot block is unlocked and can be programmed or erased. When RP# is at a logic low level the boot block is locked, the deep power-down mode is enabled and the WSM is reset preventing any blocks from being programmed or erased, therefore providing data protection during power transitions. When RP# transitions from logic low to logic high, the flash memory enters the read-array mode.
OE#	I	OUTPUT ENABLE: Gates the device's outputs through the data buffers during a read cycle. OE# is active low.
WE#	I	WRITE ENABLE: Controls writes to the Command Register and array blocks. WE# is active low. Addresses and data are latched on the rising edge of the WE# pulse.

1.4 Pin Descriptions for x8/x16 28F200BX-L (Continued)

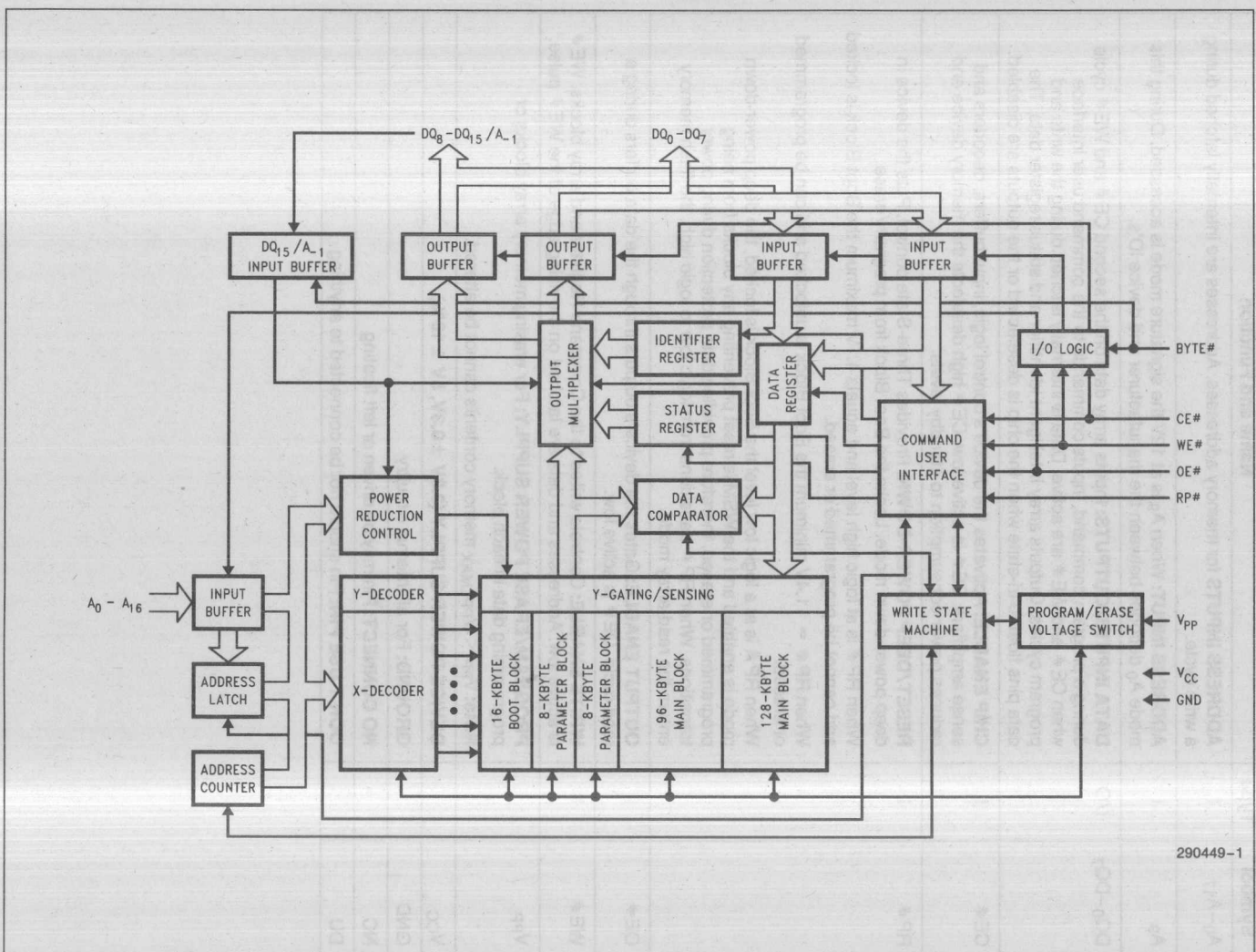
Symbol	Type	Name and Function
BYTE#	I	BYTE# ENABLE: Controls whether the device operates in the byte-wide mode (x8) or the word-wide mode (x16). BYTE# = "0" enables the byte-wide mode, where data is read and programmed on DQ ₀ –DQ ₇ and DQ ₁₅ /A _{–1} becomes the lowest order address that decodes between the upper and lower byte. DQ ₈ –DQ ₁₄ are tri-stated during the byte-wide mode. BYTE# = "1" enables the word-wide mode where data is read and programmed on DQ ₀ –DQ ₁₅ .
V _{PP}		PROGRAM/ERASE POWER SUPPLY: For erasing memory array blocks or programming data in each block. Note: V _{PP} < V _{PPLMAX} memory contents cannot be altered.
V _{CC}		DEVICE POWER SUPPLY (3.3V ± 0.3V, 5V ± 10%)
GND		GROUND: For all internal circuitry.
NC		NO CONNECT: Pin may be driven or left floating.
DU		DON'T USE PIN: Pin should not be connected to anything.

CE#	I	CHIP ENABLE: Activates the device's control logic, input buffers, decoders and sense amplifiers. CE# is active low. CE# high deasserts the memory device and reduces power consumption to standby levels. If CE# and RP# are high, but not at a CMOS high level, the standby current will increase due to current flow through the CE# and RP# input stages.
RP#	I	RESET/DEEP POWER-DOWN: Provides three-state control. The device in deep power-down mode locks the boot block from program/erase. When RP# is at logic high level and outputs A ₁₅ maximum the boot block is locked and cannot be programmed or erased. When RP# = 1.14V minimum the boot block is unlocked and can be programmed or erased. When RP# is at a logic low level the boot block is locked, the deep power-down mode is enabled and the WSM is reset preventing any blocks from being programmed or erased, therefore providing data protection during power transitions. When RP# transitions from logic low to logic high, the flash memory enters the read-array mode.
OE#	I	OUTPUT ENABLE: Gates the device's outputs through the data buffers during a read cycle. OE# is active low.
WE#	I	WRITE ENABLE: Controls writes to the Command Register and array blocks. WE# is active low. Address and data are latched on the rising edge of the WE# pulse.

1.5 Pin Descriptions for x8 28F002BX-L

Symbol	Type	Name and Function
A ₀ -A ₁₇	I	ADDRESS INPUTS for memory addresses. Addresses are internally latched during a write cycle.
A ₉	I	ADDRESS INPUT: When A ₉ is at 12V the signature mode is accessed. During this mode A ₀ decodes between the manufacturer and device ID's.
DQ ₀ -DQ ₇	I/O	DATA INPUTS/OUTPUTS: Inputs array data on the second CE# and WE# cycle during a program command. Inputs commands to the command user interface when CE# and WE# are active. Data is internally latched during the write and program cycles. Outputs array Intelligent Identifier and status register data. The data pins float to tri-state when the chip is deselected or the outputs are disabled.
CE#	I	CHIP ENABLE: Activates the device's control logic, input buffers, decoders and sense amplifiers. CE# is active low; CE# high deselects the memory device and reduces power consumption to standby levels.
RP#	I	RESET/DEEP POWER-DOWN: Provides Three-State control. Puts the device in deep power-down mode. Locks the Boot Block from program/erase. When RP# is at logic high level and equals 4.1V maximum the Boot Block is locked and cannot be programmed or erased. When RP# = 11.4V minimum the Boot Block is unlocked and can be programmed or erased. When RP# is at a logic low level the Boot Block is locked, the deep power-down mode is enabled and the WSM is reset preventing any blocks from being programmed or erased, therefore providing data protection during power transitions. When RP# transitions from logic low to logic high, the flash memory enters the read-array mode.
OE#	I	OUTPUT ENABLE: Gates the device's outputs through the data buffers during a read cycle. OE# is active low.
WE#	I	WRITE ENABLE: Controls writes to the Command Register and array blocks. WE# is active low. Addresses and data are latched on the rising edge of the WE# pulse.
V _{PP}		PROGRAM/ERASE POWER SUPPLY: For erasing memory array blocks or programming data in each block. Note: V _{PP} < V _{PPLMAX} memory contents cannot be altered.
V _{CC}		DEVICE POWER SUPPLY (3.3V ± 0.3V, 5V ± 10%)
GND		GROUND: For all internal circuitry
NC		NO CONNECT: Pin may be driven or left floating
DU		DON'T USE PIN: Pin should not be connected to anything

2.0 28F200BX-L PRODUCTS DESCRIPTION



2.1 28F200BX-L Memory Organization

2.1.1 BLOCKING

The 28F200BX-L uses a blocked array architecture to provide independent erasure of memory blocks. A block is erased independently of other blocks in the array when an address is given within the block address range and the Erase Setup and Erase Confirm commands are written to the CUI. The 28F200BX-L is a random read/write memory, only erasure is performed by block.

2.1.1.1 Boot Block Operation and Data Protection

The 16-Kbyte boot block provides a lock feature for secure code storage. The intent of the boot block is to provide a secure storage area for the kernel code that is required to boot a system in the event of power failure or other disruption during code update. This lock feature ensures absolute data integrity by preventing the boot block from being written or erased when RP# is not at 12V. The boot block can be erased and written when RP# is held at 12V for the duration of the erase or program operation. This allows customers to change the boot code when necessary while providing security when needed. See the Block Memory Map section for address locations of the boot block for the 28F200BX-TL and 28F200BX-BL.

2.1.1.2 Parameter Block Operation

The 28F200BX-L has 2 parameter blocks (8 Kbytes each). The parameter blocks are intended to provide storage for frequently updated system parameters and configuration or diagnostic information. The parameter blocks can also be used to store additional boot or main code. The parameter blocks however, do not have the hardware write protection feature that the boot block has. The parameter blocks provide for more efficient memory utilization when dealing with parameter changes versus regularly blocked devices. See the Block Memory Map section for address locations of the parameter blocks for the 28F200BX-TL and 28F200BX-BL.

2.1.1.3 Main Block Operation

Two main blocks of memory exist on the 28F200BX-L (1 x 128-Kbyte block and 1 x 96-Kbyte blocks). See the following section on Block Memory Map for the address location of these blocks for the 28F200BX-TL and 28F200BX-BL products.

2.1.2 BLOCK MEMORY MAP

Two versions of the 28F200BX-L product exist to support two different memory maps of the array blocks in order to accommodate different microprocessor protocols for boot code location. The 28F200BX-TL memory map is inverted from the 28F200BX-BL memory map.

2.1.2.1 28F200BX-BL Memory Map

The 28F200BX-BL device has the 16-Kbyte boot block located from 00000H to 01FFFFH to accommodate those microprocessors that boot from the bottom of the address map at 00000H. In the 28F200BX-BL the first 8-Kbyte parameter block resides in memory space from 02000H to 02FFFFH. The second 8-Kbyte parameter block resides in memory space from 03000H to 03FFFFH. The 96-Kbyte main block resides in memory space from 04000H to 0FFFFH. The 128-Kbyte main block resides in memory space from 10000H to 1FFFFH (word locations). See Figure 7.

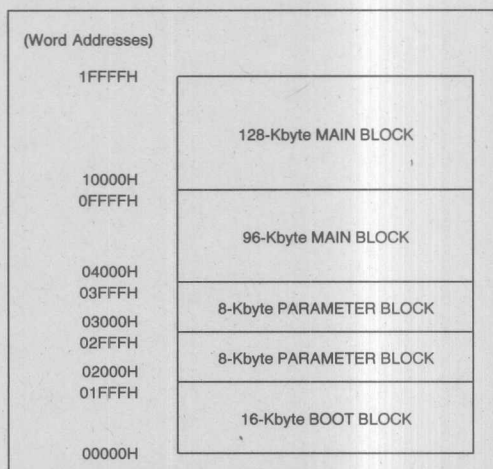


Figure 7. 28F200BX-BL Memory Map

The 28F200BX-TL device has the 16-Kbyte boot block located from 1E000H to 1FFFFH to accommodate those microprocessors that boot from the top of the address map. In the 28F200BX-TL the first 8-Kbyte parameter block resides in memory space from 1D000H to 1DFFFH. The second 8-Kbyte parameter block resides in memory space from 1C000H to 1CFFFH. The 96-Kbyte main block resides in memory space from 10000H to 1BFFFH. The 128-Kbyte main block resides in memory space from 00000H to 0FFFFH as shown below in Figure 8.

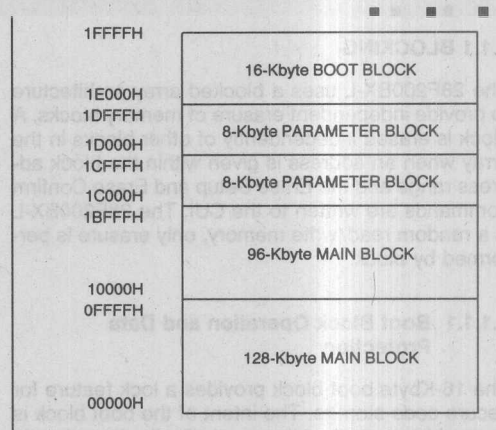


Figure 8. 28F200BX-TL Memory Map

The 28F200BX-TL device has the 16-Kbyte boot block located from 1E000H to 1FFFFH to accommodate those microprocessors that boot from the top of the address map. In the 28F200BX-TL the first 8-Kbyte parameter block resides in memory space from 1D000H to 1DFFFH. The second 8-Kbyte parameter block resides in memory space from 1C000H to 1CFFFH. The 96-Kbyte main block resides in memory space from 10000H to 1BFFFH. The 128-Kbyte main block resides in memory space from 00000H to 0FFFFH as shown below in Figure 8. (word locations). See Figure 7.

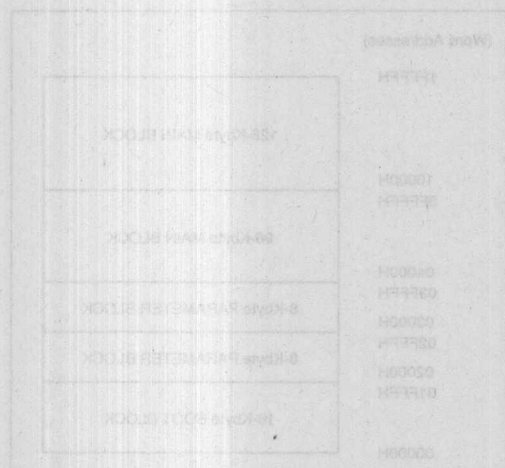
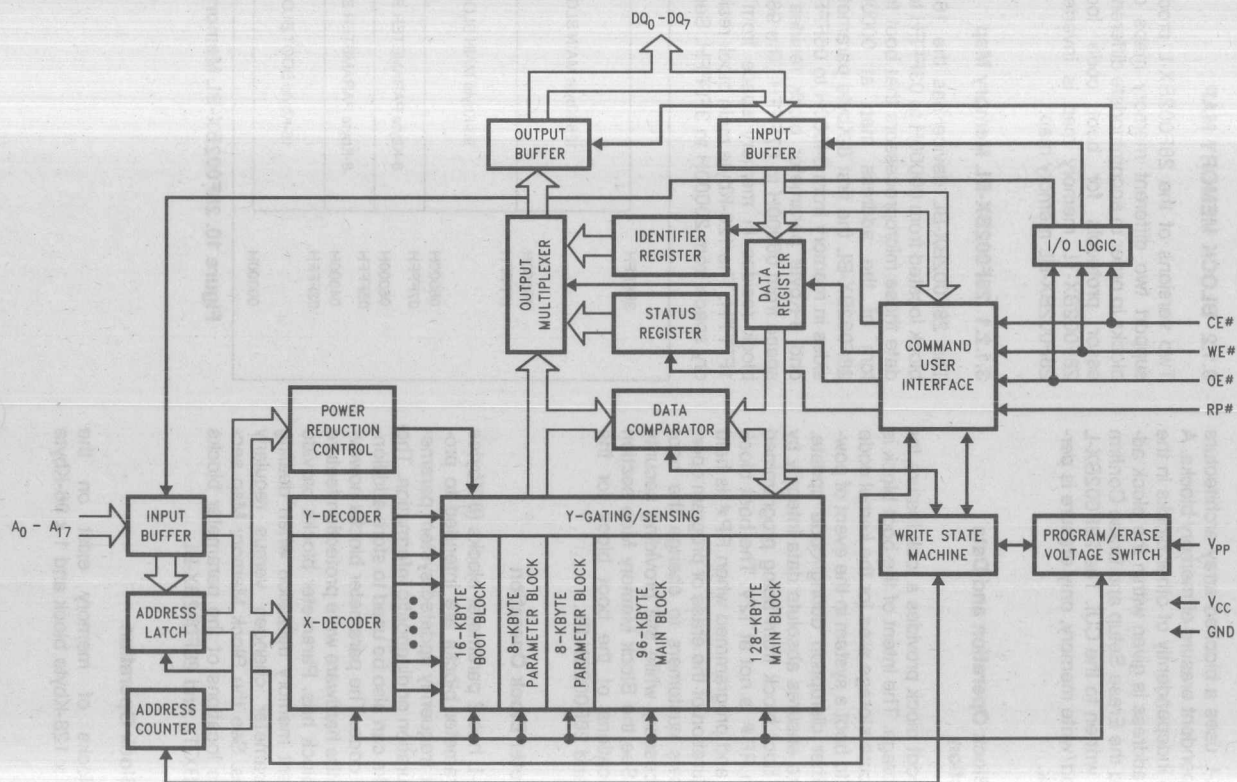


Figure 7. 28F200BX-TL Memory Map

The 28F200BX-TL has 2 parameter blocks (8 Kbytes each). The parameter blocks are intended to provide storage for frequently updated system parameters and configuration or diagnostic information. The parameter blocks can also be used to store additional data at boot or run time. The parameter blocks are not protected by hardware write protection features. The boot block has the parameter blocks and the main block. The parameter blocks are used for most system memory utilization when dealing with parameter changes versus regularly blocked devices. See the Block Memory Map section for address locations of the parameter blocks for the 28F200BX-TL and 28F200BX-BL.

Figure 9. 28F002BX-L Byte-Wide Block Diagram



290449-2

3.1 28F002BX-L Memory Organization

3.1.1 BLOCKING

The 28F002BX-L uses a blocked array architecture to provide independent erasure of memory blocks. A block is erased independently of other blocks in the array when an address is given within the block address range and the Erase Setup and Erase Confirm commands are written to the CUI. The 28F002BX-L is a random read/write memory, only erasure is performed by block.

3.1.1.1 Boot Block Operation and Data Protection

The 16-Kbyte boot block provides a lock feature for secure code storage. The intent of the boot block is to provide a secure storage area for the kernel code that is required to boot a system in the event of power failure or other disruption during code update. This lock feature ensures absolute data integrity by preventing the boot block from being programmed or erased when RP# is not at 12V. The boot block can be erased and programmed when RP# is held at 12V for the duration of the erase or program operation. This allows customers to change the boot code when necessary while still providing security when needed. See the Block Memory Map section for address locations of the boot block for the 28F002BX-TL and 28F002BX-BL.

3.1.1.2 Parameter Block Operation

The 28F002BX-L has 2 parameter blocks (8 Kbytes each). The parameter blocks are intended to provide storage for frequently updated system parameters and configuration or diagnostic information. The parameter blocks can also be used to store additional boot or main code. The parameter blocks however, do not have the hardware write protection feature that the boot block has. Parameter blocks provide for more efficient memory utilization when dealing with small parameter changes versus regularly blocked devices. See the Block Memory Map section for address locations of the parameter blocks for the 28F002BX-TL and 28F002BX-BL.

3.1.1.3 Main Block Operation

Two main blocks of memory exist on the 28F002BX-L (1 x 128-Kbyte block and 1 x 96-Kbyte block).

See the following section on Block Memory Map for the address location of these blocks for the 28F002BX-TL and 28F002BX-BL.

3.1.2 BLOCK MEMORY MAP

Two versions of the 28F002BX-L product exist to support two different memory maps of the array blocks in order to accommodate different microprocessor protocols for boot code location. The 28F002BX-TL memory map is inverted from the 28F002BX-BL memory map.

3.1.2.1 28F002BX-BL Memory Map

The 28F002BX-BL device has the 16-Kbyte boot block located from 00000H to 03FFFFH to accommodate those microprocessors that boot from the bottom of the address map at 00000H. In the 28F002BX-BL the first 8-Kbyte parameter block resides in memory from 04000H to 05FFFFH. The second 8-Kbyte parameter block resides in memory space from 06000H to 07FFFFH. The 96-Kbyte main block resides in memory space from 08000H to 1FFFFH. The 128-Kbyte main block resides in memory space from 20000H to 3FFFFH. See Figure 10.

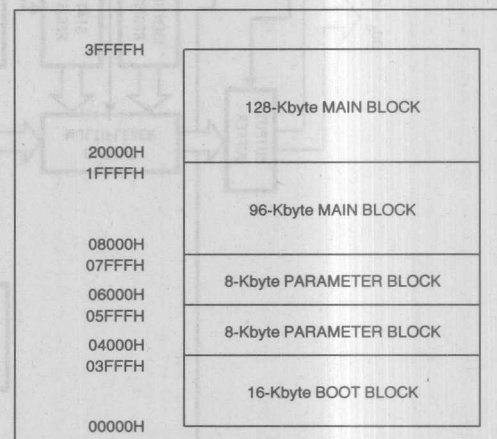


Figure 10. 28F002BX-BL Memory Map

3.1.2.2 28F002BX-TL Memory Map

The 28F002BX-TL device has the 16-Kbyte boot block located from 3C000H to 3FFFFH to accommodate those microprocessors that boot from the top of the address map. In the 28F002BX-TL the first 8-Kbyte parameter block resides in memory space from 3A000H to 3BFFFH. The second 8-Kbyte parameter block resides in memory space from 38000H to 39FFFH. The 96-Kbyte main block resides in memory space from 20000H to 37FFFH. The 128-Kbyte main block resides in memory space from 00000H to 1FFFFH.

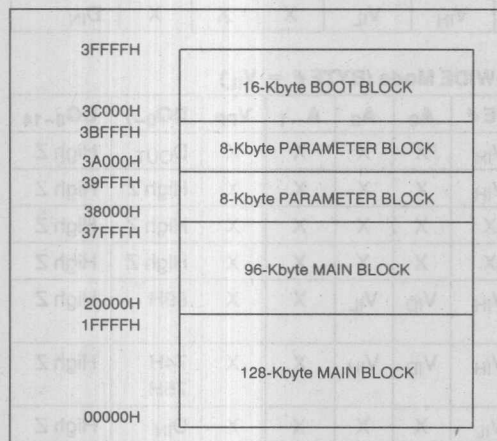


Figure 11. 28F002BX-TL Memory Map

4.0 PRODUCT FAMILY PRINCIPLES OF OPERATION

Flash memory augments EPROM functionality with in-circuit electrical write and erase. The 2-Mbit flash family utilizes a Command User Interface (CUI) and internally generated and timed algorithms to simplify write and erase operations.

The CUI allows for fixed power supplies during erasure and programming, and maximum EPROM compatibility.

In the absence of high voltage on the V_{pp} pin, the 2-Mbit flash family will only successfully execute the following commands: Read Array, Read Status Register, Clear Status Register and Intelligent Identifier mode. The device provides standard EPROM read, standby and output disable operations. Manufacturer Identification and Device Identification data can be accessed through the CUI or through the standard EPROM A9 high voltage access (V_{ID}) (for PROM programmer equipment).

The same EPROM read, standby and output disable functions are available when high voltage is applied to the V_{pp} pin. In addition, high voltage on V_{pp} allows write and erase of the device. All functions associated with altering memory contents: write and erase, Intelligent Identifier read and Read Status are accessed via the CUI.

The purpose of the Write State Machine (WSM) is to completely automate the write and erasure of the device. The WSM will begin operation upon receipt of a signal from the CUI and will report status back through a Status Register. The CUI will handle the $WE\#$ interface to the data and address latches, as well as system software requests for status while the WSM is in operation.

4.1 28F200BX-L Bus Operations

Flash memory reads, erases and writes in-system via the local CPU. All bus cycles to or from the flash memory conform to standard microprocessor bus cycles.

Table 1. Bus Operations for WORD-WIDE Mode (BYTE# = V_{IH})

Mode	Notes	RP#	CE#	OE#	WE#	A ₉	A ₀	V _{PP}	DQ ₀₋₁₅
Read	1, 2, 3	V _{IH}	V _{IL}	V _{IL}	V _{IH}	X	X	X	D _{OUT}
Output Disable		V _{IH}	V _{IL}	V _{IH}	V _{IH}	X	X	X	High Z
Standby		V _{IH}	V _{IH}	X	X	X	X	X	High Z
Deep Power-Down	9	V _{IL}	X	X	X	X	X	X	High Z
Intelligent Identifier (Mfr)	4	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IL}	X	0089H
Intelligent Identifier (Device)	4, 5, 10	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IH}	X	2274H 2275H
Write	6, 7, 8	V _{IH}	V _{IL}	V _{IH}	V _{IL}	X	X	X	D _{IN}

Table 2. Bus Operations for BYTE-WIDE Mode (BYTE# = V_{IL})

Mode	Notes	RP#	CE#	OE#	WE#	A ₉	A ₀	A ₋₁	V _{PP}	DQ ₀₋₇	DQ ₈₋₁₄
Read	1, 2, 3	V _{IH}	V _{IL}	V _{IL}	V _{IH}	X	X	X	X	D _{OUT}	High Z
Output Disable		V _{IH}	V _{IL}	V _{IH}	V _{IH}	X	X	X	X	High Z	High Z
Standby		V _{IH}	V _{IH}	X	X	X	X	X	X	High Z	High Z
Deep Power-Down	9	V _{IL}	X	X	X	X	X	X	X	High Z	High Z
Intelligent Identifier (Mfr)	4	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IL}	X	X	89H	High Z
Intelligent Identifier (Device)	4, 5	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IH}	X	X	74H 75H	High Z
Write	6, 7, 8	V _{IH}	V _{IL}	V _{IH}	V _{IL}	X	X	X	X	D _{IN}	High Z

NOTES:

1. Refer to DC Characteristics.
2. X can be V_{IL} or V_{IH} for control pins and addresses, V_{PP}L or V_{PP}H for V_{PP}.
3. See DC characteristics for V_{PP}L, V_{PP}H, V_{HH}, V_{ID} voltages.
4. Manufacturer and Device codes may also be accessed via a CPU write sequence. A₁-A₁₆ = V_{IL}.
5. Device ID = 2274H for 28F200BX-TL and 2275H for 28F200BX-BL.
6. Refer to Table 4 for valid D_{IN} during a write operation.
7. Command writes for Block Erase or Word/Byte Write are only executed when V_{PP} = V_{PP}H.
8. To write or erase the boot block, hold RP# at V_{HH}.
9. RP# must be at GND ±0.2V to meet the 1.2 μA maximum deep power-down current.
10. The device ID codes are identical to those of the 28F2100BX 5V versions.

4.2 28F002BX-L Bus Operations

Table 3. Bus Operations

Mode	Notes	RP #	CE #	OE #	WE #	A ₉	A ₀	V _{PP}	DQ ₀₋₇
Read	1, 2, 3	V _{IH}	V _{IL}	V _{IL}	V _{IH}	X	X	X	D _{OUT}
Output Disable		V _{IH}	V _{IL}	V _{IH}	V _{IH}	X	X	X	High Z
Standby		V _{IH}	V _{IH}	X	X	X	X	X	High Z
Deep Power-Down	9	V _{IL}	X	X	X	X	X	X	High Z
Intelligent Identifier (Mfr)	4	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IL}	X	89H
Intelligent Identifier (Device)	4, 5	V _{IH}	V _{IL}	V _{IL}	V _{IH}	V _{ID}	V _{IH}	X	7CH 7DH
Write	6, 7, 8	V _{IH}	V _{IL}	V _{IH}	V _{IL}	X	X	X	D _{IN}

NOTES:

1. Refer to DC Characteristics.
2. X can be V_{IL} or V_{IH} for control pins and addresses, V_{PP}L or V_{PP}H for V_{PP}.
3. See DC characteristics for V_{PP}L, V_{PP}H, V_{HH}, V_{ID} voltages.
4. Manufacturer and Device codes may also be accessed via a CUI write sequence. A₁-A₁₇ = V_{IL}.
5. Device ID = 7CH for 28F002BX-TL and 7DH for 28F002BX-BL.
6. Refer to Table 4 for valid D_{IN} during a write operation.
7. Command writes for Block erase or byte program are only executed when V_{PP} = V_{PP}H.
8. Program or erase the Boot block by holding RP# at V_{HH}.
9. RP# must be at GND ± 0.2V to meet the 1.2 μA maximum deep power-down current.
10. The device ID codes are identical to those of the 28F002BX 5V versions.

4.3 Read Operations

The 2-Mbit flash family has three user read modes; Array, Intelligent Identifier, and Status Register. Status Register read mode will be discussed in detail in the "Write Operations" section.

During power-up conditions (V_{CC} supply ramping), it takes a maximum of 700 ns from V_{CC} at 3.0V minimum to obtain valid data on the outputs.

4.3.1 READ ARRAY

If the memory is not in the Read Array mode, it is necessary to write the appropriate read mode command to the CUI. The 2-Mbit flash family has three control functions, all of which must be logically active, to obtain data at the outputs. Chip-Enable CE# is the device selection control. Power-Down RP# is the device power control. Output-Enable OE# is the DATA INPUT/OUTPUT (DQ[0:15] or DQ[0:7]) direction control and when active is used to drive data from the selected memory on to the I/O bus.

4.3.1.1 Output Control

With OE# at logic-high level (V_{IH}), the output from the device is disabled and data input/output pins (DQ[0:15] or DQ[0:7]) are tri-stated. Data input is then controlled by WE#.

4.3.1.2 Input Control

With WE# at logic-high level (V_{IH}), input to the device is disabled. Data Input/Output pins (DQ[0:15] or DQ[0:7]) are controlled by OE#.

4.3.2 INTELLIGENT IDENTIFIERS

28F200BX-L Products

The manufacturer and device codes are read via the CUI or by taking the A₉ pin to 12V. Writing 90H to the CUI places the device into Intelligent Identifier read mode. A read of location 00000H outputs the manufacturer's identification code, 0089H, and location 00001H outputs the device code; 2274H for 28F200BX-TL, 2275H for 28F200BX-BL. When BYTE# is at a logic low only the lower byte of the above signatures is read and DQ₁₅/A₋₁ is a "don't care" during Intelligent Identifier mode. A read array command must be written to the CUI to return to the read array mode.

28F002BX-L Products

The manufacturer and device codes are also read via the CUI or by taking the A_9 pin to 12V. Writing 90H to the CUI places the device into Intelligent Identifier read mode. A read of location 00000H outputs the manufacturer's identification code, 89H, and location 00001H outputs the device code; 7CH for 28F002BX-TL, 7DH for 28F002BX-BL.

4.4 Write Operations

Commands are written to the CUI using standard microprocessor write timings. The CUI serves as the interface between the microprocessor and the internal chip operation. The CUI can decipher Read Array, Read Intelligent Identifier, Read Status Register, Clear Status Register, Erase and Program commands. In the event of a read command, the CUI simply points the read path at either the array, the Intelligent Identifier, or the status register depending on the specific read command given. For a program or erase cycle, the CUI informs the write state machine that a write or erase has been requested. During a program cycle, the Write State Machine will control the program sequences and the CUI will only respond to status reads. During an erase cycle, the CUI will respond to status reads and erase suspend. After the Write State Machine has completed its task, it will allow the CUI to respond to its full command set. The CUI will stay in the current command state until the microprocessor issues another command.

The CUI will successfully initiate an erase or write operation only when V_{PP} is within its voltage range. Depending upon the application, the system designer may choose to make the V_{PP} power supply switchable, available only when memory updates are desired. The system designer can also choose to "hard-wire" V_{PP} to 12V. The 2-Mbit flash family is designed to accommodate either design practice. It is recommended that RP# be tied to logical Reset for data protection during unstable CPU reset function as described in the "Product Family Overview" section.

4.4.1 BOOT BLOCK WRITE OPERATIONS

In the case of Boot Block modifications (write and erase), RP# is set to $V_{HH} = 12V$ typically, in addition to V_{PP} at high voltage. However, if RP# is not at V_{HH} when a program or erase operation of the boot block is attempted, the corresponding status register bit (Bit 4 for Program and Bit 5 for Erase, refer to Table 5 for Status Register Definitions) is set to indicate the failure to complete the operation.

4.4.2 COMMAND USER INTERFACE (CUI)

The Command User Interface (CUI) serves as the interface to the microprocessor. The CUI points the read/write path to the appropriate circuit block as described in the previous section. After the WSM has completed its task, it will set the WSM Status bit to a "1", which will also allow the CUI to respond to its full command set. Note that after the WSM has returned control to the CUI, the CUI will remain in its current state.

4.4.2.1 Command Set

Command Codes	Device Mode
00	Invalid/Reserved
10	Alternate Program Setup
20	Erase Setup
40	Program Setup
50	Clear Status Register
70	Read Status Register
90	Intelligent Identifier
B0	Erase Suspend
D0	Erase Resume/Erase Confirm
FF	Read Array

4.4.2.2 Command Function Descriptions

Device operations are selected by writing specific commands into the CUI. Table 4 defines the 2-Mbit flash family commands.

Table 4. Command Definitions

Command	Bus Cycles Req'd	Notes	First Bus Cycle			Second Bus Cycle		
		8	Operation	Address	Data	Operation	Address	Data
Read Array	1	1	Write	X	FFH			
Intelligent Identifier	3	2, 4	Write	X	90H	Read	IA	IID
Read Status Register	2	3	Write	X	70H	Read	X	SRD
Clear Status Register	1		Write	X	50H			
Erase Setup/Erase Confirm	2	5	Write	BA	20H	Write	BA	D0H
Word/Byte Write Setup/Write	2	6, 7	Write	WA	40H	Write	WA	WD
Erase Suspend/Erase Resume	2		Write	X	B0H	Write	X	D0H
Alternate Word/Byte Write Setup/Write	2	2, 3, 7	Write	WA	10H	Write	WA	WD

NOTES:

1. Bus operations are defined in Tables 1, 2, 3.
 2. IA = Identifier Address: 00H for manufacturer code, 01H for device code.
 3. SRD = Data read from Status Register.
 4. IID = Intelligent Identifier Data.
- Following the Intelligent Identifier Command, two read operations access manufacturer and device codes.
5. BA = Address within the block being erased.
 6. WA = Address to be written.
 - WD = Data to be written at location WA.
 7. Either 40H or 10H commands is valid.
 8. When writing commands to the device, the upper data bus [DQ₈-DQ₁₅] = X (28F200BX-L-only) which is either V_{CC} or V_{SS} to avoid burning additional current.

Invalid/Reserved

These are unassigned commands. It is not recommended that the customer use any command other than the valid commands specified above. Intel reserves the right to redefine these codes for future functions.

Read Array (FFH)

This single write command points the read path at the array. If the host CPU performs a CE#/OE# controlled read immediately following a two-write sequence that started the WSM, then the device will output status register contents. If the Read Array command is given after Erase Setup the device is reset to read the array. A two Read Array command sequence (FFH) is required to reset to Read Array after Program Setup.

Intelligent Identifier (90H)

After this command is executed, the CUI points the output path to the Intelligent Identifier circuits. Only Intelligent Identifier values at addresses 0 and 1 can be read (only address A0 is used in this mode, all other address inputs are ignored).

Read Status Register (70H)

This is one of the two commands that is executable while the state machine is operating. After this command is written, a read of the device will output the contents of the status register, regardless of the address presented to the device.

The device automatically enters this mode after program or erase has completed.

Clear Status Register (50H)

The WSM can only set the Program Status and Erase Status bits in the status register, it can not clear them. Two reasons exist for operating the status register in this fashion. The first is a synchronization. The WSM does not know when the host CPU has read the status register, therefore it would not know when to clear the status bits. Secondly, if the CPU is programming a string of bytes, it may be more efficient to query the status register after programming the string. Thus, if any errors exist while programming the string, the status register will return the accumulated error status.

This command simply sets the CUI into a state such that the next write will load the address and data registers. Either 40H or 10H can be used for Program Setup. Both commands are included to accommodate efforts to achieve an industry standard command code set.

Program

The second write after the program setup command, will latch addresses and data. Also, the CUI initiates the WSM to begin execution of the program algorithm. While the WSM finishes the algorithm, the device will output Status Register contents. Note that the WSM cannot be suspended during programming.

Erase Setup (20H)

Prepares the CUI for the Erase Confirm command. No other action is taken. If the next command is not an Erase Confirm command then the CUI will set both the Program Status and Erase Status bits of the Status Register to a "1", place the device into the Read Status Register state, and wait for another command.

Erase Confirm (D0H)

If the previous command was an Erase Setup command, then the CUI will enable the WSM to erase, at the same time closing the address and data latches, and respond only to the Read Status Register and Erase Suspend commands. While the WSM is executing, the device will output Status Register data when OE# is toggled low. Status Register data can only be updated by toggling either OE# or CE# low.

Erase Suspend (B0H)

This command only has meaning while the WSM is executing an Erase operation, and therefore will only be responded to during an erase operation. After this command has been executed, the CUI will initiate the WSM to suspend Erase operations, and then return to responding to only Read Status Register or to the Erase Resume commands. Once the WSM has reached the Suspend state, it will set an output into the CUI which allows the CUI to respond to the Read Array, Read Status Register, and Erase Resume commands. In this mode, the CUI will not respond to any other commands. The WSM will also

continue to run, idling in the SUSPEND state, regardless of the state of all input control pins, with the exclusion of RP#. RP# low will immediately shut down the WSM and the remainder of the chip.

Erase Resume (D0H)

This command will cause the CUI to clear the Suspend state and set the WSM Status bit to a "0", but only if an Erase Suspend command was previously issued. Erase Resume will not have any effect in all other conditions.

4.4.3 STATUS REGISTER

The 2-Mbit flash family contains a status register which may be read to determine when a program or erase operation is complete, and whether that operation completed successfully. The status register may be read at any time by writing the Read Status command to the CUI. After writing this command, all subsequent Read operations output data from the status register until another command is written to the CUI. A Read Array command must be written to the CUI to return to the Read Array mode.

The status register bits are output on DQ[0:7] whether the device is in the byte-wide (x8) or word-wide (x16) mode for the 28F200BX-L. In the word-wide mode the upper byte, DQ[8:15] is set to 00H during a Read Status command. In the byte-wide mode, DQ[8:14] are tri-stated and DQ_{15/A-1} retains the low order address function.

It should be noted that the contents of the status register are latched on the falling edge of OE# or CE# whichever occurs last in the read cycle. This prevents possible bus errors which might occur if the contents of the status register change while reading the status register. CE# or OE# must be toggled with each subsequent status read, or the completion of a program or erase operation will not be evident.

The Status Register is the interface between the microprocessor and the Write State Machine (WSM). When the WSM is active, this register will indicate the status of the WSM, and will also hold the bits indicating whether or not the WSM was successful in performing the desired operation. The WSM sets status bits "Three" through "Seven" and clears bits "Six" and "Seven", but cannot clear status bits "Three" through "Five". These bits can only be cleared by the controlling CPU through the use of the Clear Status Register command.

4.4.3.1 Status Register Bit Definition

Table 5. Status Register Definitions

WSMS	ESS	ES	PS	VPPS	R	R	R
7	6	5	4	3	2	1	0
<p>SR.7 = WRITE STATE MACHINE STATUS 1 = Ready 0 = Busy</p> <p>SR.6 = ERASE SUSPEND STATUS 1 = Erase Suspend 0 = Erase in Progress/Completed</p> <p>SR.5 = ERASE STATUS 1 = Error in Block Erasure 0 = Successful Block Erase</p> <p>SR.4 = PROGRAM STATUS 1 = Error in Byte/Word Program 0 = Successful Byte/Word Program</p> <p>SR.3 = V_{pp} STATUS 1 = V_{pp} Low Detect; Operation Abort 0 = V_{pp} OK</p> <p>SR.2–SR.0 = RESERVED FOR FUTURE ENHANCEMENTS These bits are reserved for future use and should be masked out when polling the Status Register.</p>				<p>NOTES:</p> <p>Write State Machine Status bit must first be checked to determine byte/word program or block erase completion, before the Program or Erase Status bits are checked for success.</p> <p>When Erase Suspend is issued, WSM halts execution and sets both WSMS and ESS bits to "1". ESS bit remains set to "1" until an Erase Resume command is issued.</p> <p>When this bit is set to "1", WSM has applied the maximum number of erase pulses to the block and is still unable to successfully perform an erase verify.</p> <p>When this bit is set to "1", WSM has attempted but failed to Program a byte or word.</p> <p>The V_{pp} Status bit unlike an A/D converter, does not provide continuous indication of V_{pp} level. The WSM interrogates the V_{pp} level only after the byte write or block erase command sequences have been entered and informs the system if V_{pp} has not been switched on. The V_{pp} Status bit is not guaranteed to report accurate feedback between V_{ppL} and V_{ppH}.</p>			

4

4.4.3.2 Clearing the Status Register

Certain bits in the status register are set by the write state machine, and can only be reset by the system software. These bits can indicate various failure conditions. By allowing the system software to control the resetting of these bits, several operations may be performed (such as cumulatively programming several bytes or erasing multiple blocks in sequence). The status register may then be read to determine if an error occurred during that programming or erasure series. This adds flexibility to the way the device may be programmed or erased. To clear the status register, the Clear Status Register command is written to the CUI. Then, any other command may be issued to the CUI. Note again that before a read cycle can be initiated, a Read Array command must be written to the CUI to specify whether the read data is to come from the array, status register, or Intelligent Identifier.

4.4.4 PROGRAM MODE

Program is executed by a two-write sequence. The Program Setup command is written to the CUI followed by a second write which specifies the address and data to be programmed. The write state machine will execute a sequence of internally timed events to:

1. program the desired bits of the addressed memory word (byte), and
2. verify that the desired bits are sufficiently programmed.

Programming of the memory results in specific bits within a byte or word being changed to a "0".

If the user attempts to program "1"s, there will be no change of the memory cell content and no error occurs.

Similar to erasure, the status register indicates whether programming is complete. While the program sequence is executing, bit 7 of the status register is a "0". The status register can be polled by toggling either CE# or OE# to determine when the program sequence is complete. Only the Read Status Register command is valid while programming is active.

When programming is complete, the status bits, which indicate whether the program operation was successful, should be checked. If the programming operation was unsuccessful, Bit 4 of the status register is set to a "1" to indicate a Program Failure. If Bit 3 is set then V_{PP} was not within acceptable limits, and the WSM will not execute the programming sequence.

The status register should be cleared before attempting the next operation. Any CUI instruction can follow after programming is completed; however, it must be recognized that reads from the memory, status register, or Intelligent Identifier cannot be accomplished until the CUI is given the appropriate command. A Read Array command must first be given before memory contents can be read.

Figure 12 shows a system software flowchart for device byte programming operation. Figure 13 shows a similar flowchart for device word programming operation (28F200BX-L-only).

4.4.5 ERASE MODE

Erasure of a single block is initiated by writing the Erase Setup and Erase Confirm commands to the CUI, along with the addresses, A[12:16] for the 28F200BX-L or A[12:17] for the 28F002BX-L, identifying the block to be erased. These addresses are latched internally when the Erase Confirm command is issued. Block erasure results in all bits within the block being set to "1".

The WSM will execute a sequence of internally timed events to:

1. program all bits within the block
2. verify that all bits within the block are sufficiently programmed
3. erase all bits within the block and
4. verify that all bits within the block are sufficiently erased

While the erase sequence is executing, Bit 7 of the status register is a "0".

When the status register indicates that erasure is complete, the status bits, which indicate whether the erase operation was successful, should be checked.

If the erasure operation was unsuccessful, Bit 5 of the status register is set to a "1" to indicate an Erase Failure. If V_{PP} was not within acceptable limits after the Erase Confirm command is issued, the WSM will not execute an erase sequence; instead, Bit 5 of the status register is set to a "1" to indicate an Erase Failure, and Bit 3 is set to a "1" to identify that V_{PP} supply voltage was not within acceptable limits.

The status register should be cleared before attempting the next operation. Any CUI instruction can follow after erasure is completed; however, it must be recognized that reads from the memory array, status register, or Intelligent Identifier can not be accomplished until the CUI is given the appropriate command. A Read Array command must first be given before memory contents can be read.

Figure 14 shows a system software flowchart for Block Erase operation.

4.4.5.1 Suspending and Resuming Erase

Since an erase operation typically requires 2 to 5 seconds to complete, an Erase Suspend command is provided. This allows erase-sequence interruption in order to read data from another block of the memory. Once the erase sequence is started, writing the Erase Suspend command to the CUI requests that the Write State Machine (WSM) pause the erase sequence at a predetermined point in the erase algorithm. The status register must be read to determine when the erase operation has been suspended.

At this point, a Read Array command can be written to the CUI in order to read data from blocks other than that which is being suspended. The only other valid command at this time is the Erase Resume command or Read Status Register operation.

Figure 15 shows a system software flowchart detailing the operation.

During Erase Suspend mode, the chip can go into a pseudo-standby mode by taking CE# to V_{IH} and the active current is now a maximum of 6 mA. If the chip is enabled while in this mode by taking CE# to V_{IL}, the Erase Resume command can be issued to resume the erase operation.

Upon completion of reads from any block other than the block being erased, the Erase Resume command must be issued. When the Erase Resume command is given, the WSM will continue with the erase sequence and complete erasing the block. As with the end of erase, the status register must be read, cleared, and the next instruction issued in order to continue.

4.4.6 EXTENDED CYCLING

Intel has designed extended cycling capability into its ETOX III flash memory technology. The 2-Mbit low voltage flash family is designed for 10,000 pro-

gram/erase cycles on each of the five blocks. The combination of low electric fields, clean oxide processing and minimized oxide area per memory cell subjected to the tunneling electric field, results in very high cycling capability.

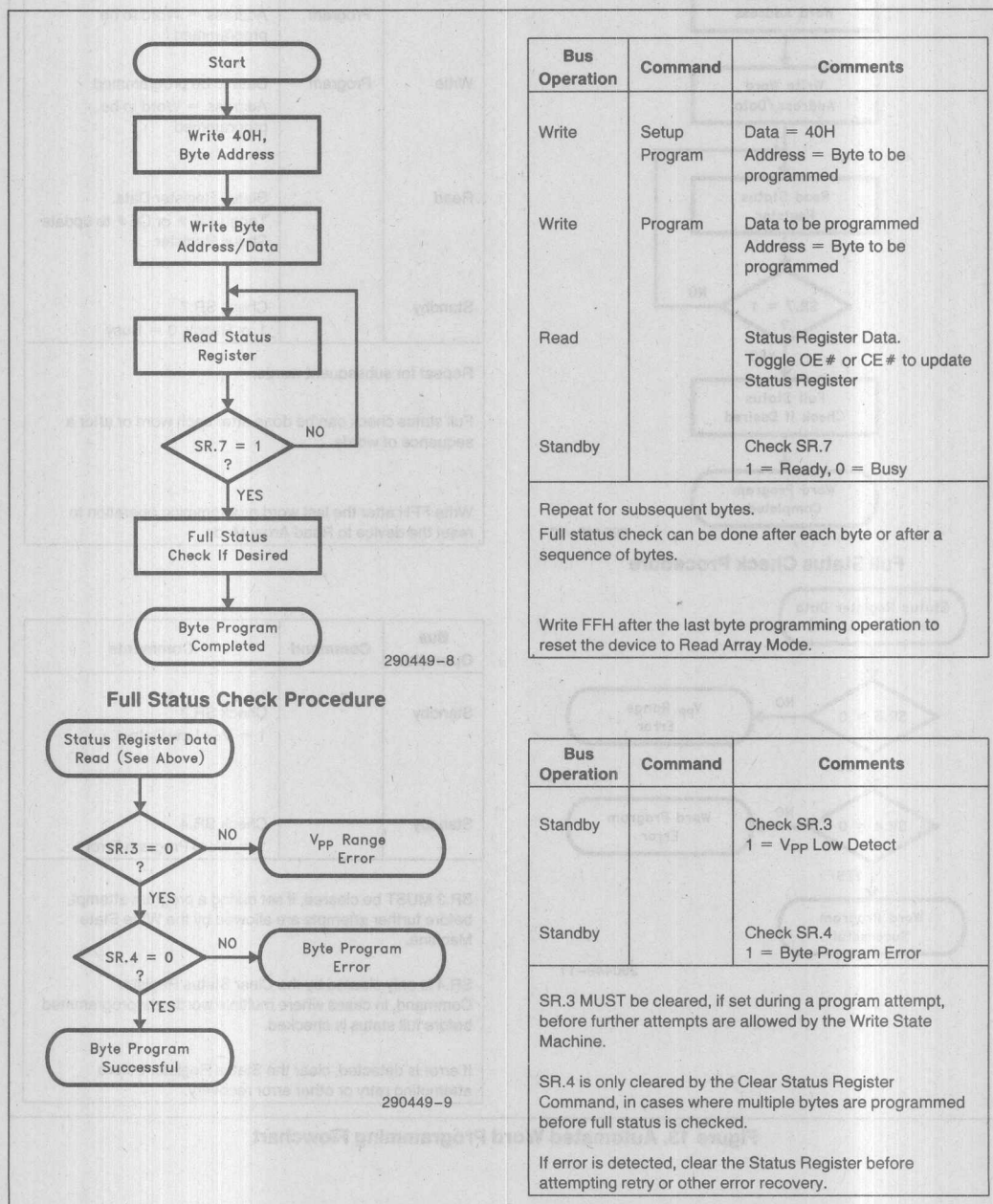


Figure 12. Automated Byte Programming Flowchart

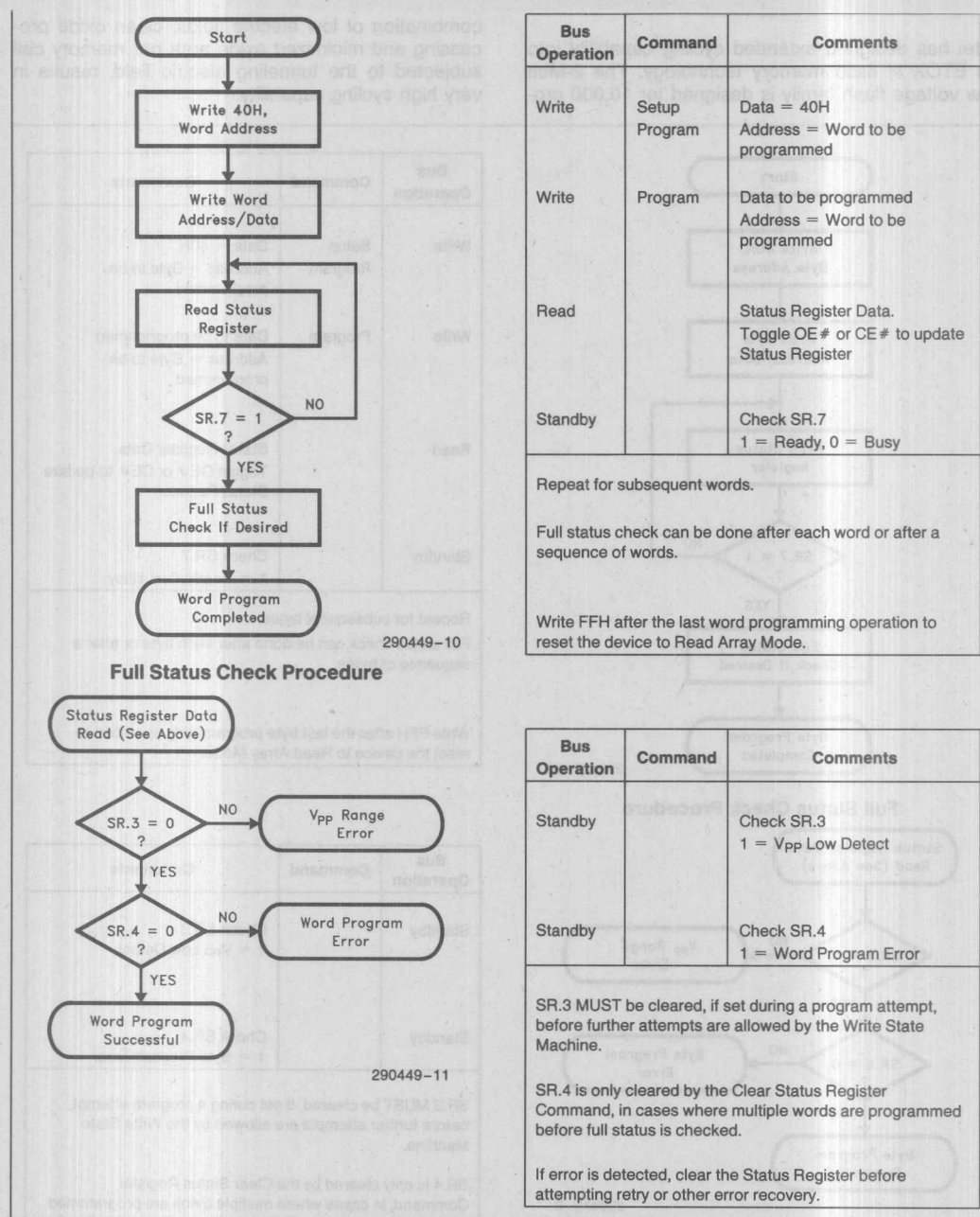


Figure 13. Automated Word Programming Flowchart

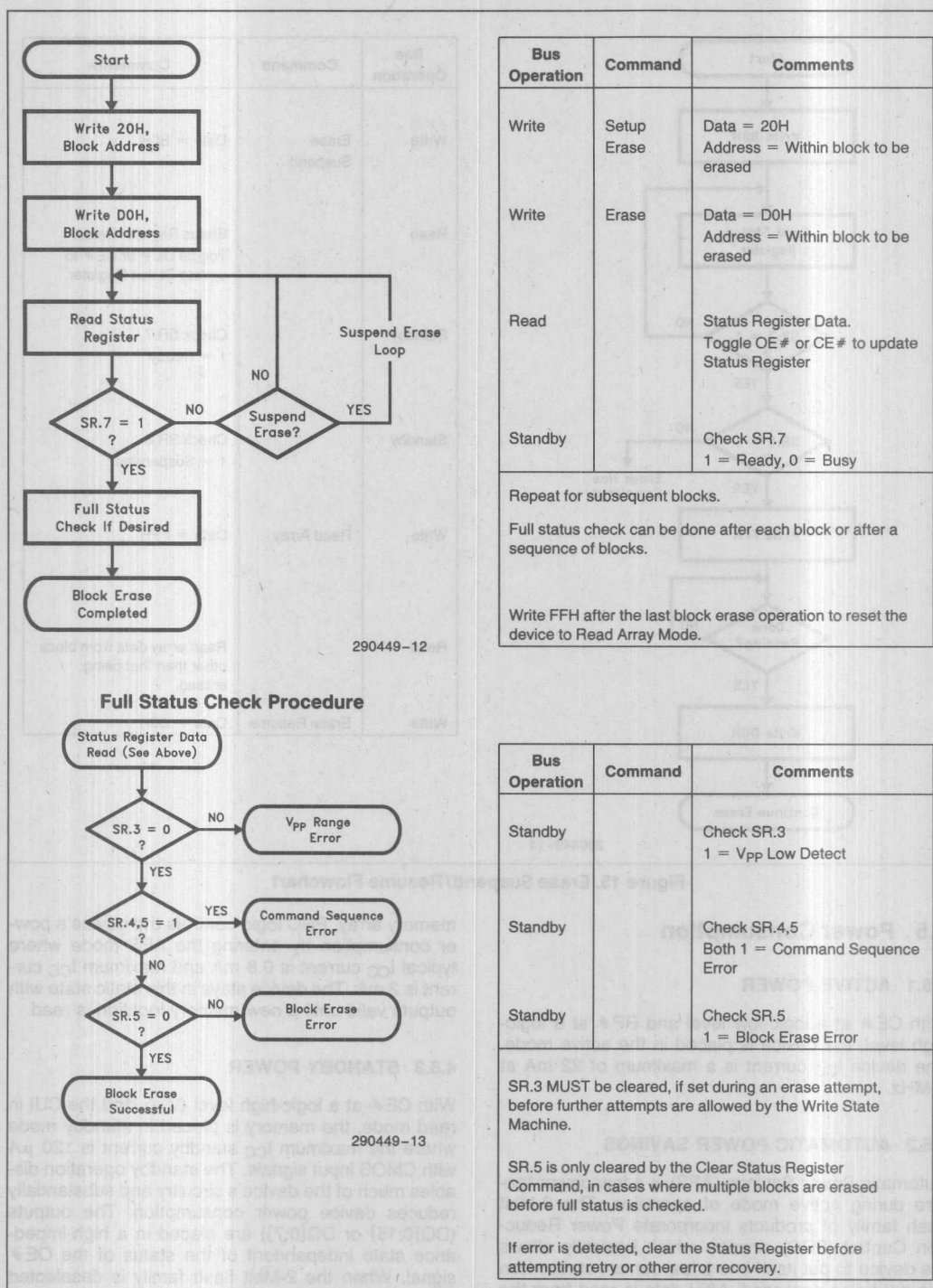


Figure 14. Automated Block Erase Flowchart

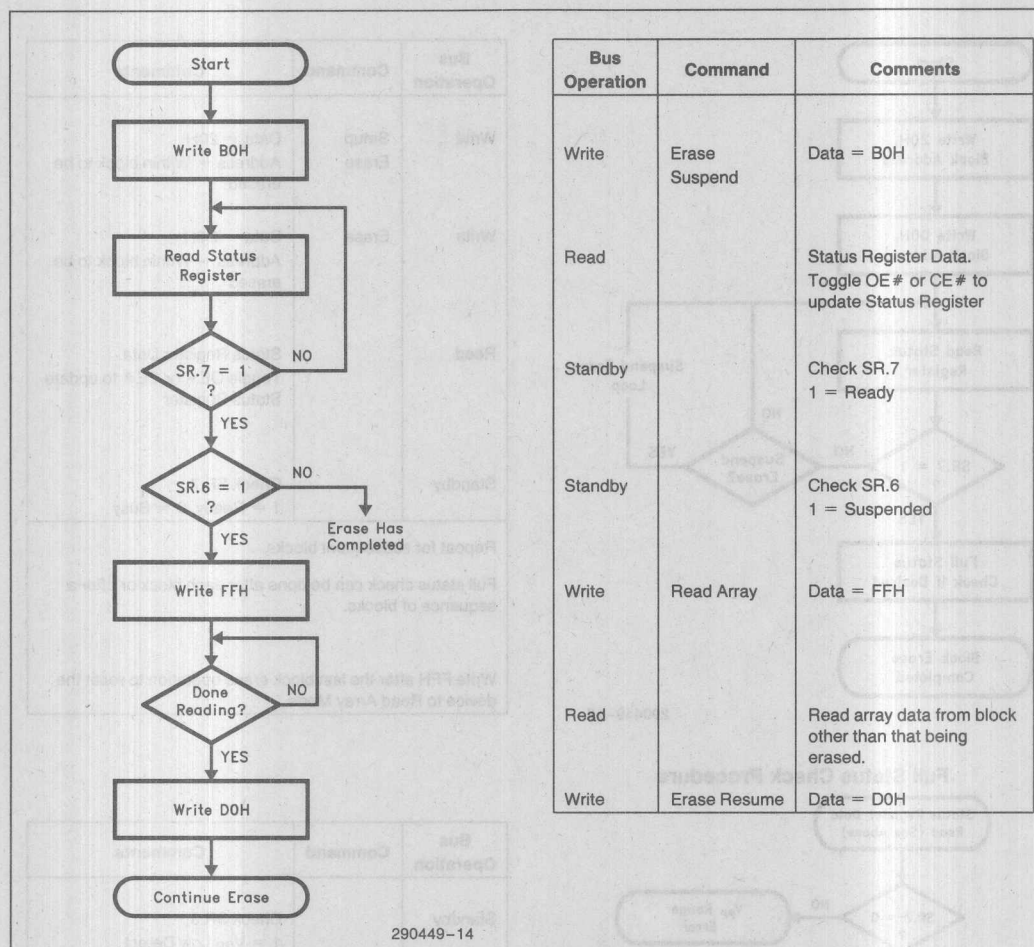


Figure 15. Erase Suspend/Resume Flowchart

4.5 Power Consumption

4.5.1 ACTIVE POWER

With CE# at a logic-low level and RP# at a logic-high level, the device is placed in the active mode. The device I_{CC} current is a maximum of 22 mA at 5 MHz.

4.5.2 AUTOMATIC POWER SAVINGS

Automatic Power Savings (APS) is a low power feature during active mode of operation. The 2-Mbit flash family of products incorporate Power Reduction Control (PRC) circuitry which basically allows the device to put itself into a low current state when it is not being accessed. After data is read from the

memory array, PRC logic controls the device's power consumption by entering the APS mode where typical I_{CC} current is 0.8 mA and maximum I_{CC} current is 2 mA. The device stays in this static state with outputs valid until a new memory location is read.

4.5.3 STANDBY POWER

With CE# at a logic-high level (V_{IH}), and the CUI in read mode, the memory is placed in standby mode where the maximum I_{CC} standby current is 120 μA with CMOS input signals. The standby operation disables much of the device's circuitry and substantially reduces device power consumption. The outputs (DQ[0:15] or DQ[0:7]) are placed in a high-impedance state independent of the status of the OE# signal. When the 2-Mbit flash family is deselected during erase or program functions, the devices will

continue to perform the erase or program function and consume program or erase active power until program or erase is completed.

4.5.4 RESET/DEEP POWER-DOWN

The 2-Mbit flash family supports a typical I_{CC} of 0.2 μA in deep power-down mode. One of the target markets for these devices is in portable equipment where the power consumption of the machine is of prime importance. The 2-Mbit flash family has a $RP\#$ pin which places the device in the deep power-down mode. When $RP\#$ is at a logic-low ($GND \pm 0.2V$), all circuits are turned off and the device typically draws 0.2 μA of V_{CC} current.

During read modes, the $RP\#$ pin going low deselects the memory and places the output drivers in a high impedance state. Recovery from the deep power-down state, requires a minimum of 700 ns to access valid data (t_{PHQV}).

During erase or program modes, $RP\#$ low will abort either erase or program operation. The contents of the memory are no longer valid as the data has been corrupted by the $RP\#$ function. As in the read mode above, all internal circuitry is turned off to achieve the 0.2 μA current level.

$RP\#$ transitions to V_{IL} or turning power off to the device will clear the status register.

The use of $RP\#$ during system reset is important with automated write/erase devices. When the system comes out of reset, it expects to read from the flash memory. Automated flash memories provide status information when accessed during write/erase modes. If a CPU reset occurs with no flash memory reset, proper CPU initialization would not occur because the flash memory would be providing the status information instead of array data. Intel's Flash Memories allow proper CPU initialization following a system reset through the use of $RP\#$ input. In this application, $RP\#$ is controlled by the same $RESET\#$ signal that resets the system CPU.

4.6 Power-Up Operation

The 2-Mbit flash family is designed to offer protection against accidental block erasure or programming during power transitions. Upon power-up the 2-Mbit flash family is indifferent as to which power supply, V_{PP} or V_{CC} , powers-up first. Power supply sequencing is not required.

The 2-Mbit flash family ensures the CUI is reset to the read mode on power-up.

In addition, on power-up the user must either drop $CE\#$ low or present a new address to ensure valid data at the outputs.

A system designer must guard against spurious writes for V_{CC} voltages above V_{LKO} when V_{PP} is active. Since both $WE\#$ and $CE\#$ must be low for a command write, driving either signal to V_{IH} will inhibit writes to the device. The CUI architecture provides an added level of protection since alteration of memory contents can only occur after successful completion of the two-step command sequences. Finally the device is disabled until $RP\#$ is brought to V_{IH} , regardless of the state of its control inputs. This feature provides yet another level of memory protection.

4.7 Power Supply Decoupling

Flash memory's power switching characteristics require careful device decoupling methods. System designers are interested in 3 supply current issues:

- Standby current levels (I_{CCS})
- Active current levels (I_{CCR})
- Transient peaks produced by falling and rising edges of $CE\#$.

Transient current magnitudes depend on the device outputs' capacitive and inductive loading. Two-line control and proper decoupling capacitor selection will suppress these transient voltage peaks. Each flash device should have a 0.1 μF ceramic capacitor connected between each V_{CC} and GND , and between its V_{PP} and GND . These high frequency, low-inductance capacitors should be placed as close as possible to the package leads.

4.7.1 V_{PP} TRACE ON PRINTED CIRCUIT BOARDS

Writing to flash memories while they reside in the target system, requires special consideration of the V_{PP} power supply trace by the printed circuit board designer. The V_{PP} pin supplies the flash memory cell's current for programming and erasing. One should use similar trace widths and layout considerations given to the V_{CC} power supply trace. Adequate V_{PP} supply traces and decoupling will decrease spikes and overshoots.

4.7.2 V_{CC} , V_{PP} AND $RP\#$ TRANSITIONS

The CUI latches commands as issued by system software and is not altered by V_{PP} or $CE\#$ transitions or WSM actions. Its state upon power-up, after exit from deep power-down mode or after V_{CC} transitions below V_{LKO} (Lockout voltage), is Read Array mode.

After any word/byte write or block erase operation is complete and even after V_{PP} transitions down to V_{PPL} , the CUI must be reset to Read Array mode via the Read Array command when accesses to the flash memory are desired.

Absolute Maximum Ratings

Operating Temperature	
During Read	0°C to 70°C(1)
During Block Erase and Word/Byte Write	0°C to 70°C
Temperature Under Bias	-10°C to +80°C
Storage Temperature	-65°C to +125°C
Voltage on Any Pin	
(except V _{CC} , V _{PP} , A ₉ and RP#)	
with Respect to GND	-2.0V to +7.0V(2)
Voltage on Pin RP# or Pin A ₉	
with Respect to GND	-2.0V to 13.5V(2, 3)
V _{PP} Program Voltage with Respect to GND during Block Erase and Word/Byte Write	-2.0V to +14.0V(2, 3)
V _{CC} Supply Voltage	
with Respect to GND	-2.0V to +7.0V(2)
Output Short Circuit Current	100 mA(4)

NOTES:

- Operating temperature is for commercial product defined by this specification.
- Minimum DC voltage is -0.5V on input/output pins. During transitions, this level may undershoot to -2.0V for periods <20 ns. Maximum DC voltage on input/output pins is V_{CC} + 0.5V which, during transitions, may overshoot to V_{CC} + 2.0V for periods <20 ns.
- Maximum DC voltage on V_{PP} may overshoot to +14.0V for periods <20 ns. Maximum DC voltage on RP# or A₉ may overshoot to 13.5V for periods <20 ns.
- Output shorted for no more than one second. No more than one output shorted at a time.
- AC Specifications are valid at both voltage ranges. See DC Characteristics table for voltage range-specific specifications.

OPERATING CONDITIONS

Symbol	Parameter	Notes	Min	Max	Unit
T _A	Operating Temperature		0	70	°C
V _{CC}	V _{CC} Supply Voltage	5	3.00	3.60	V
V _{CC}	V _{CC} Supply Voltage	5	4.50	5.50	V

DC CHARACTERISTICS V_{CC} = 3.3V ± 0.3V

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Condition
I _{LI}	Input Load Current	1			±1.0	μA	V _{CC} = V _{CC} Max V _{IN} = V _{CC} or GND
I _{LO}	Output Leakage Current	1			±10	μA	V _{CC} = V _{CC} Max V _{OUT} = V _{CC} or GND
I _{CCS}	V _{CC} Standby Current	1, 3		45	120	μA	V _{CC} = V _{CC} Max CE# = RP# = V _{CC} ± 0.2V
				45	120	μA	V _{CC} = V _{CC} Max CE# = RP# = V _{IH}
I _{CCD}	V _{CC} Deep Power-down Current	1	0.20	1.2		μA	RP# = GND ± 0.2V

NOTICE: This data sheet contains preliminary information on new products in production. The specifications are subject to change without notice. Verify with your local Intel Sales office that you have the latest data sheet before finalizing a design.

***WARNING:** Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.

DC CHARACTERISTICS $V_{CC} = 3.3V \pm 0.3V$ (Continued)

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Condition
I _{CCR}	V _{CC} Read Current for 28F200BX-L Word-Wide and Byte-Wide Mode and 28F002BX-L Byte-Wide Mode	1, 5, 6		15	25	mA	V _{CC} = V _{CC} Max, CE# = GND f = 5 MHz, I _{OUT} = 0 mA CMOS Inputs
					25	mA	V _{CC} = V _{CC} Max, CE# = V _{IL} f = 5 MHz, I _{OUT} = 0 mA TTL Inputs
I _{CCW}	V _{CC} Word/Byte Write Current	1, 4			30	mA	Word/Byte Write in Progress
I _{CCE}	V _{CC} Block Erase Current	1, 4			20	mA	Block Erase in Progress
I _{CCES}	V _{CC} Erase Suspend Current	1, 2	3		6	mA	Block Erase Suspended, CE# = V _{IH}
I _{PPS}	V _{PP} Standby Current	1			±15	μA	V _{PP} ≤ V _{CC}
I _{PPD}	V _{PP} Deep Power-down Current	1			5.0	μA	RP# = GND ±0.2V
I _{PPR}	V _{PP} Read Current	1			200	μA	V _{PP} > V _{CC}
I _{PPW}	V _{PP} Word Write Current	1			40	mA	V _{PP} = V _{PPH} Word Write in Progress
I _{PPW}	V _{PP} Byte Write Current	1			30	mA	V _{PP} = V _{PPH} Byte Write in Progress
I _{PPE}	V _{PP} Block Erase Current	1			30	mA	V _{PP} = V _{PPH} Block Erase in Progress
I _{PPES}	V _{PP} Erase Suspend Current	1			200	μA	V _{PP} = V _{PPH} Block Erase Suspended
I _{RP#}	RP# Boot Block Unlock Current	1, 4			500	μA	RP# = V _{HH}
I _{ID}	A _g Intelligent Identifier Current	1, 4			500	μA	A _g = V _{ID}
V _{ID}	A _g Intelligent Identifier Voltage		11.4	12.0	13.0	V	
V _{IL}	Input Low Voltage		-0.5		0.6	V	
V _{IH}	Input High Voltage		2.0		V _{CC} + 0.5	V	
V _{OL}	Output Low Voltage				0.4	V	V _{CC} = V _{CC} Min I _{OL} = 2 mA
V _{OH}	Output High Voltage		2.4			V	V _{CC} = V _{CC} Min I _{OH} = -2 mA
V _{PPL}	V _{PP} during Normal Operations	3	0.0		4.1	V	
V _{PPH}	V _{PP} during Erase/Write Operations		11.4	12.0	12.6	V	
V _{LKO}	V _{CC} Erase/Write Lock Voltage		2.0			V	
V _{HH}	RP# Unlock Voltage		11.4	12.0	13.0	V	Boot Block Write/Erase

CAPACITANCE⁽⁴⁾ $T_A = 25^\circ\text{C}$, $f = 1\text{ MHz}$

Symbol	Parameter	Typ	Max	Unit	Condition
C_{IN}	Input Capacitance	6	8	pF	$V_{IN} = 0V$
C_{OUT}	Output Capacitance	10	12	pF	$V_{OUT} = 0V$

NOTES:

1. All currents are in RMS unless otherwise noted. Typical values at $V_{CC} = 3.3V$, $V_{PP} = 12.0V$, $T = 25^\circ\text{C}$. These currents are valid for all product versions (packages and speeds).
2. I_{CCES} is specified with the device deselected. If the device is read while in Erase Suspend Mode, current draw is the sum of I_{CCES} and I_{CCR} .
3. Block Erases and Word/Byte Writes are inhibited when $V_{PP} = V_{PPL}$ and not guaranteed in the range between V_{PPH} and V_{PPL} .
4. Sampled, not 100% tested.
5. Automatic Power Savings (APS) reduces I_{CCR} to less than 2 mA in static operation.
6. CMOS Inputs are either $V_{CC} \pm 0.2V$ or $GND \pm 0.2V$. TTL Inputs are either V_{IL} or V_{IH} .

DC CHARACTERISTICS $V_{CC} = 5.0V \pm 10\%$ ⁽⁴⁾

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Condition
I_{LI}	Input Load Current	1			± 1.0	μA	$V_{CC} = V_{CC\text{ Max}}$ $V_{IN} = V_{CC}$ or GND
I_{LO}	Output Leakage Current	1			± 10	μA	$V_{CC} = V_{CC\text{ Max}}$ $V_{OUT} = V_{CC}$ or GND
I_{CCS}	V_{CC} Standby Current				1.5	mA	$V_{CC} = V_{CC\text{ Max}}$ $CE\# = RP\# = V_{IH}$
					100	μA	$V_{CC} = V_{CC\text{ Max}}$ $CE\# = RP\# = V_{CC} \pm 0.2V$
I_{CCD}	V_{CC} Deep Power-down Current	1			1.2	μA	$RP\# = GND \pm 0.2V$
I_{CCR}	V_{CC} Read Current for 28F200BX-L Word-Wide and Byte-Wide Mode and 28F002BX-L Byte-Wide Mode	1			40	mA	$V_{CC} = V_{CC\text{ Max}}$, $CE\# = GND$ $f = 5\text{ MHz}$, $I_{OUT} = 0\text{ mA}$ CMOS Inputs
					40	mA	$V_{CC} = V_{CC\text{ Max}}$, $CE\# = V_{IL}$ $f = 5\text{ MHz}$, $I_{OUT} = 0\text{ mA}$ TTL Inputs
I_{CCW}	V_{CC} Word-Byte Write Current	1			70	mA	Word or Byte Write in Progress
I_{CCE}	V_{CC} Block Erase Current	1			30	mA	Block Erase in Progress
I_{CCES}	V_{CC} Erase Suspend Current	1, 2			10	mA	Block Erase Suspended, $CE\# = V_{IH}$
I_{PPS}	V_{PP} Standby Current	1			± 15	μA	$V_{PP} \leq V_{CC}$
I_{PPD}	V_{PP} Deep Power-down Current	1			5.0	μA	$RP\# = GND \pm 0.2V$

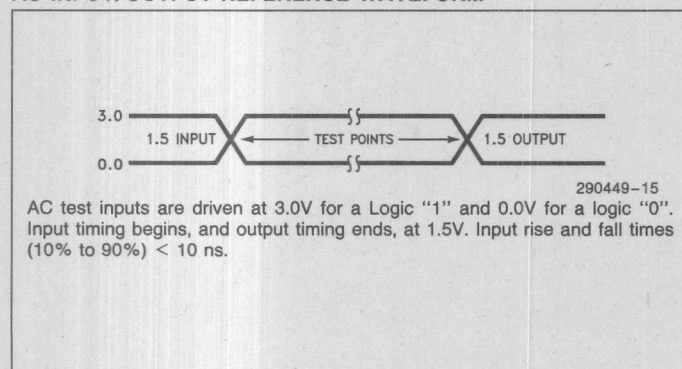
DC CHARACTERISTICS $V_{CC} = 5.0V \pm 10\%$ (Continued)

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Condition
I _{PPR}	V _{PP} Read Current	1			200	μA	V _{PP} > V _{CC}
I _{PPW}	V _{PP} Word Write Current	1			40	mA	V _{PP} = V _{PPH} Word Write in Progress
I _{PPW}	V _{PP} Byte Write Current	1			30	mA	V _{PP} = V _{PPH} Byte Write in Progress
I _{PPE}	V _{PP} Block Erase Current	1			30	mA	V _{PP} = V _{PPH} Block Erase in Progress
I _{PPES}	V _{PP} Erase Suspend Current	1			200	μA	V _{PP} = V _{PPH} Block Erase Suspended
I _{RP#}	RP# Boot Block Unlock Current	1, 4			500	μA	RP# = V _{HH}
I _{ID}	A ₉ Intelligent Identifier Current	1, 4			500	μA	A ₉ = V _{ID}
V _{ID}	A ₉ Intelligent Identifier Voltage		11.4	12.0	13.0	V	
V _{IL}	Input Low Voltage		-0.5		0.8	V	
V _{IH}	Input High Voltage		2.0		V _{CC} + 0.5	V	
V _{OL}	Output Low Voltage				0.45	V	V _{CC} = V _{CC} Min I _{OL} = 5.8 mA
V _{OH}	Output High Voltage		2.4			V	V _{CC} = V _{CC} Min I _{OH} = -2.5 mA
V _{PPL}	V _{PP} during Normal Operations	3	0.0		6.5	V	
V _{PPH}	V _{PP} during Erase/Write Operations		11.4	12.0	12.6	V	
V _{LKO}	V _{CC} Erase/Write Lock Voltage		2.2			V	
V _{HH}	RP# Unlock Voltage		11.4	12.0	13.0	V	Boot Block Write/Erase

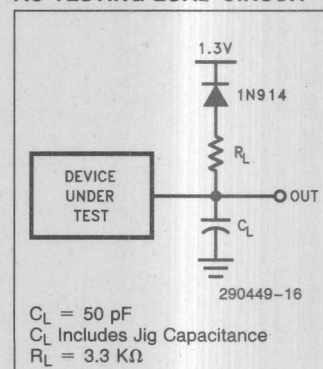
NOTES:

1. All currents are in RMS unless otherwise noted. Typical values at V_{CC} = 5.0V, V_{PP} = 12.0V, T = 25°C. These currents are valid for all product versions (packages and speeds).
2. I_{CCES} is specified with the device deselected. If the device is read while in Erase Suspend Mode, current draw is the sum of I_{CCES} and I_{CCR}.
3. Block Erase/Byte Writes are inhibited when V_{PP} = V_{PPL} and not guaranteed in the range between V_{PPH} and V_{PPL}.
4. All parameters are sampled, not 100% tested.

AC INPUT/OUTPUT REFERENCE WAVEFORM



AC TESTING LOAD CIRCUIT

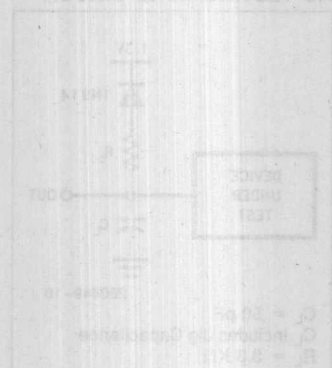


Versions			28F200BX-L150 28F002BX-L150		Unit
Symbol	Parameter	Notes	Min	Max	
t_{AVAV}	t_{RC}	Read Cycle Time	150		ns
t_{AVQV}	t_{ACC}	Address to Output Delay		150	ns
t_{ELQV}	t_{CE}	CE # to Output Delay	2	150	ns
t_{PHQV}	t_{PWH}	RP # High to Output Delay		600	ns
t_{GLQV}	t_{OE}	OE # to Output Delay	2	65	ns
t_{ELQX}	t_{LZ}	CE # to Output Low Z	3	0	ns
t_{EHQZ}	t_{HZ}	CE # High to Output High Z	3	55	ns
t_{GLQX}	t_{OLZ}	OE # to Output Low Z	3	0	ns
t_{GHQZ}	t_{DF}	OE # High to Output High Z	3	45	ns
	t_{OH}	Output Hold from Addresses, CE # or OE # Change, Whichever is First	3	0	ns
t_{ELFL} t_{ELFH}	CE # to BYTE # Switching Low to High	3		5	ns
t_{FHQV}	BYTE # Switching High to Valid Output Delay	3, 4		150	ns
t_{FLQZ}	BYTE # Switching Low to Output High Z	3		45	ns

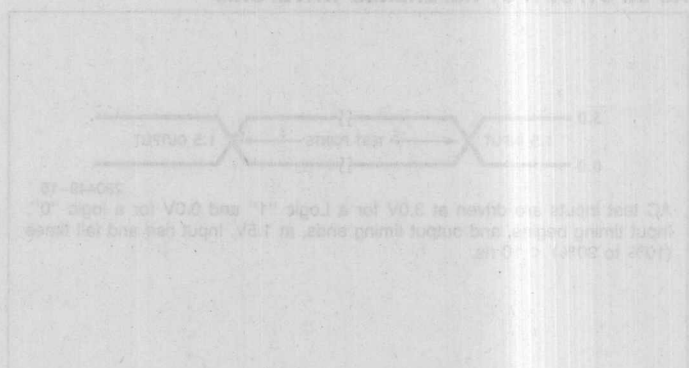
NOTES:

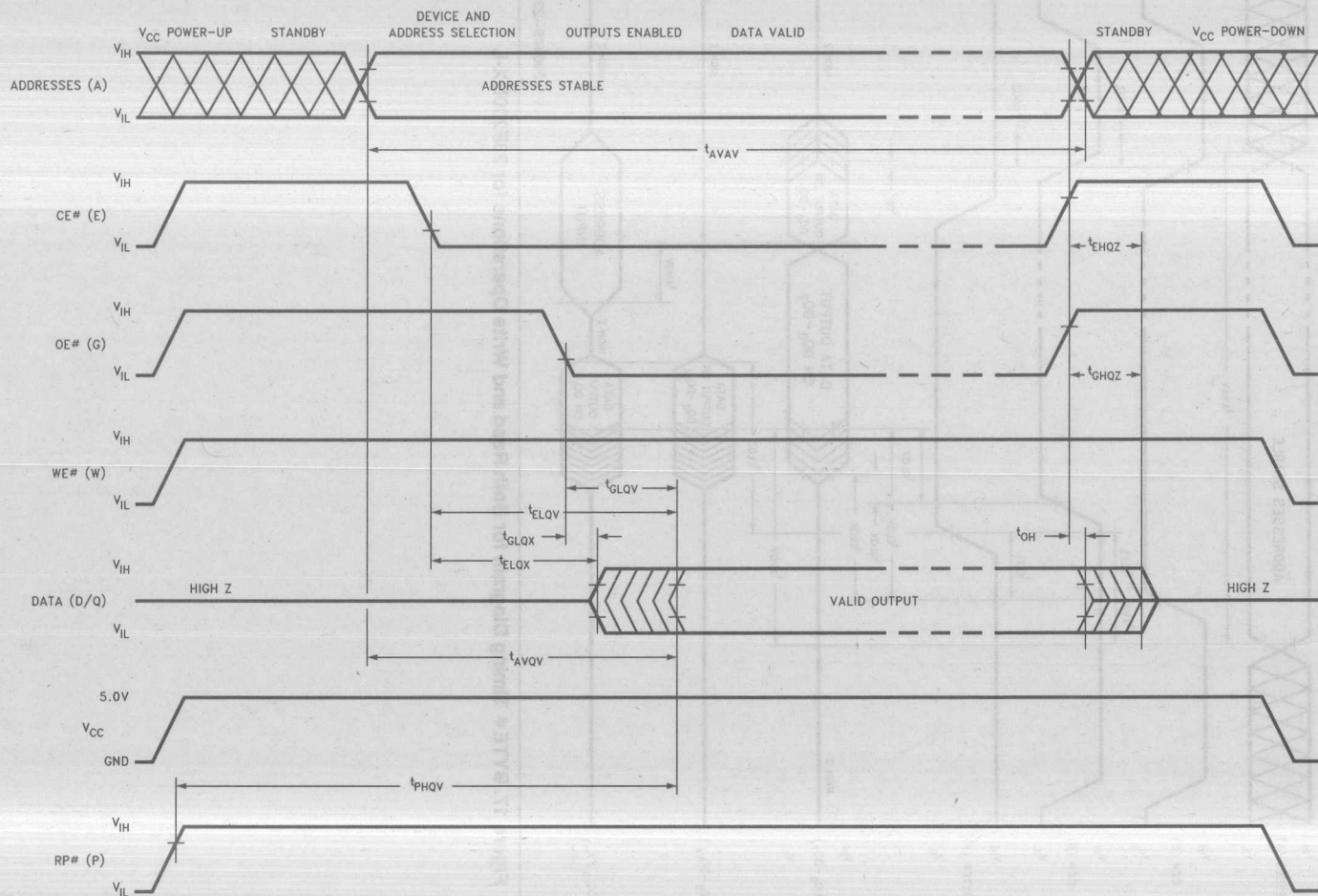
1. See AC Input/Output Reference Waveform for timing measurements.
2. OE # may be delayed up to $t_{CE} - t_{OE}$ after the falling edge of CE # without impact on t_{CE} .
3. Sampled, not 100% tested.
4. t_{FLQV} , BYTE # switching low to valid output delay will be equal to t_{AVQV} , measured from the time DQ₁₅/A₋₁ becomes valid.

AC TESTING LOAD CIRCUIT



AC INPUT/OUTPUT REFERENCE WAVEFORM





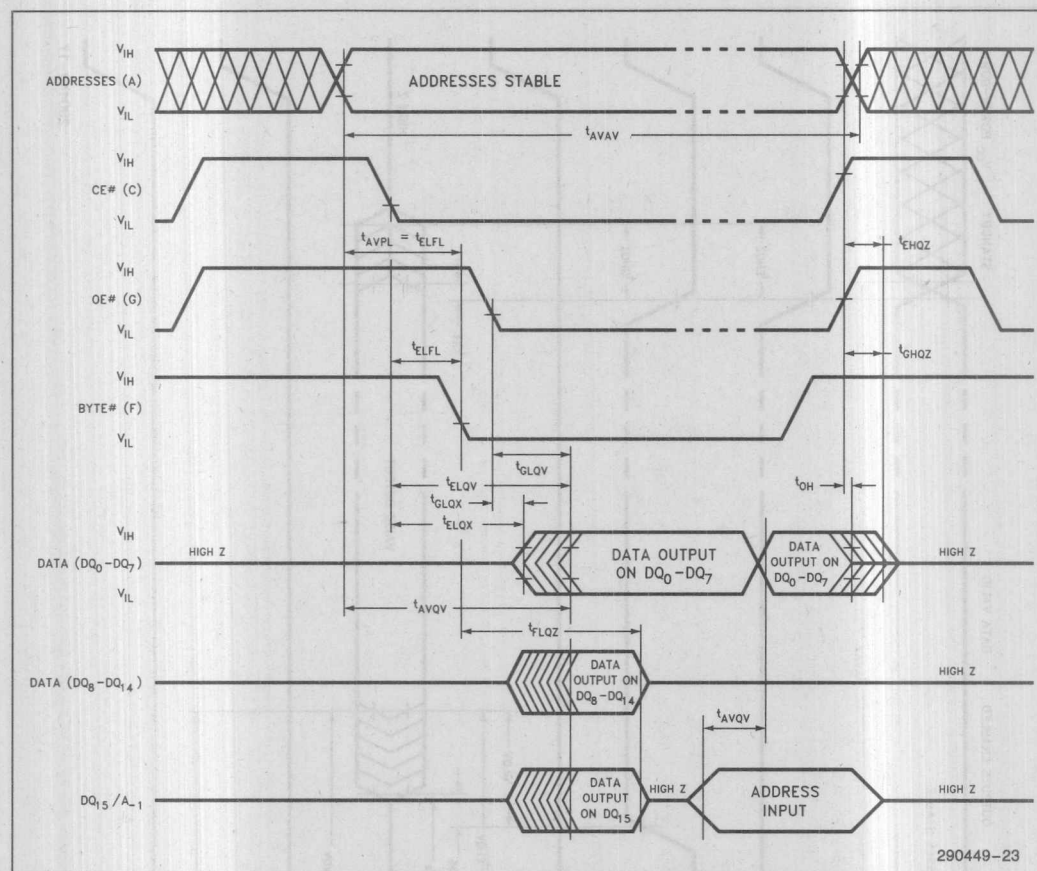


Figure 17. BYTE # Timing Diagram for Both Read and Write Operations for 28F200BX-L

AC CHARACTERISTICS For WE# Controlled Write Operations⁽¹⁾ $V_{CC} = 3.3V \pm 0.3V, 5.0V \pm 10\%$

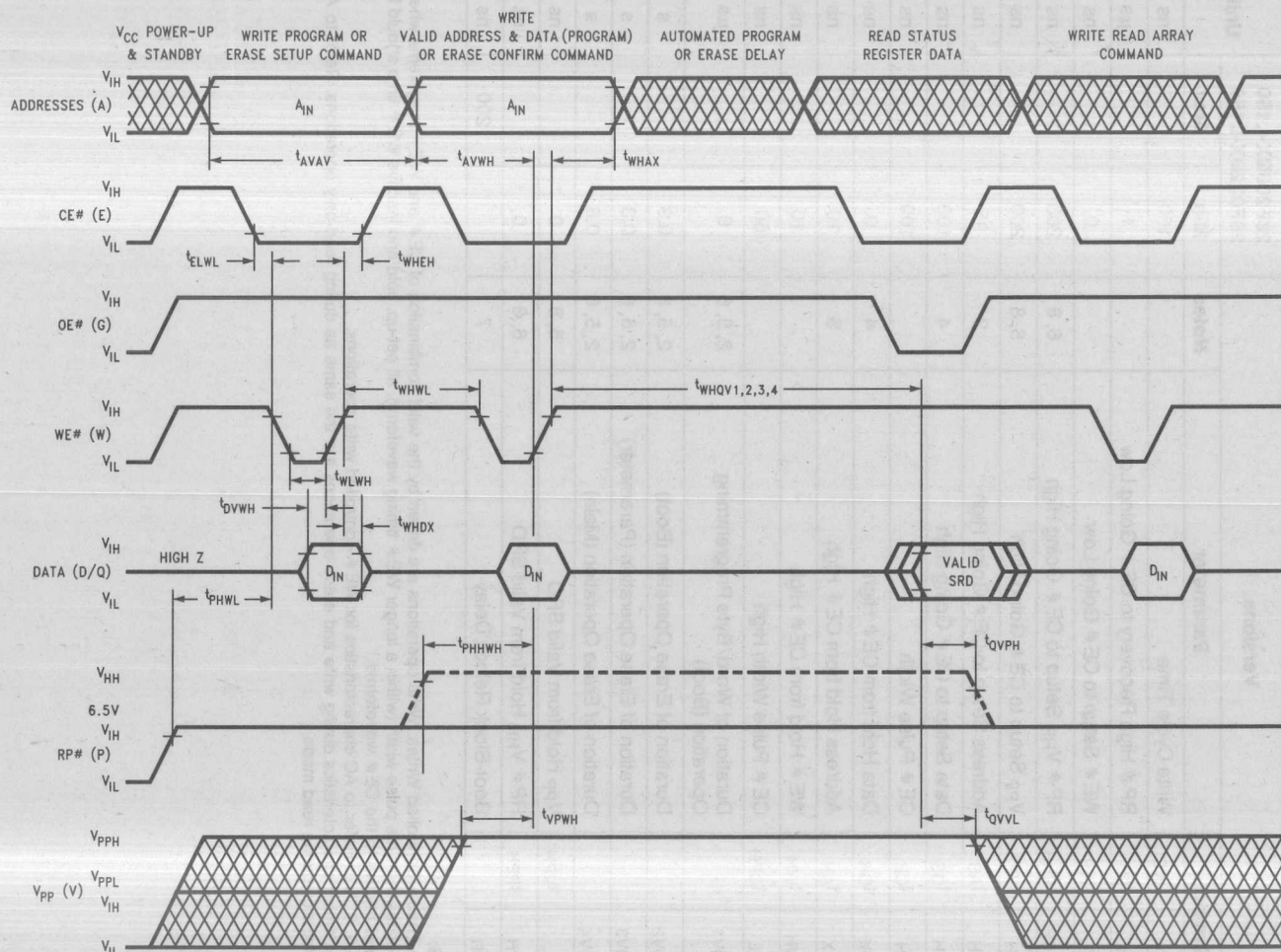
		Versions ⁽⁴⁾			28F200BX-L150 28F002BX-L150	Unit
Symbol		Parameter	Notes	Min	Max	
t _{AVAV}	t _{WC}	Write Cycle Time		150		ns
t _{PHWL}	t _{PS}	RP# High Recovery to WE# Going Low		1		μs
t _{ELWL}	t _{CS}	CE# Setup to WE# Going Low		0		ns
t _{PHHWH}	t _{PHS}	RP# V _{HH} Setup to WE# Going High	6, 8	200		ns
t _{VPWH}	t _{VPS}	V _{PP} Setup to WE# Going High	5, 8	200		ns
t _{AVWH}	t _{AS}	Address Setup to WE# Going High	3	95		ns
t _{DVWH}	t _{DS}	Data Setup to WE# Going High	4	100		ns
t _{WLWH}	t _{WP}	WE# Pulse Width		100		ns
t _{WHDX}	t _{DH}	Data Hold from WE# High	4	0		ns
t _{WHAX}	t _{AH}	Address Hold from WE# High	3	10		ns
t _{WHEH}	t _{CH}	CE# Hold from WE# High		10		ns
t _{WHWL}	t _{WPH}	WE# Pulse Width High		50		ns
t _{WHQV1}		Duration of Programming Operation (Boot)	2, 5, 6	6		μs
t _{WHQV2}		Duration of Word/Byte Programming Operation	2, 5, 6	0.3		s
t _{WHQV3}		Duration of Erase Operation (Parameter)	2, 5, 6	0.3		s
t _{WHQV4}		Duration of Erase Operation (Main)	2, 5, 6	0.6		s
t _{QWL}	t _{VPH}	V _{PP} Hold from Valid SRD	5, 8	0		ns
t _{QVPH}	t _{PHH}	RP# V _{HH} Hold from Valid SRD	6, 8	0		ns
t _{PHBR}		Boot-Block Belock Delay	7, 8		200	ns

NOTES:

1. Read timing characteristics during write and erase operations are the same as during read-only operations. Refer to AC characteristics during Read Mode.
2. The on-chip WSM completely automates program/erase operations; program/erase algorithms are now controlled internally which includes verify and margining operations.
3. Refer to command definition table for valid A_{IN}.
4. Refer to command definition table for valid D_{IN}.
5. Program/Erase durations are measured to valid SRD data (successful operation, SR.7 = 1).
6. For Boot Block Program/Erase, RP# should be held at V_{HH} until operation completes successfully.
7. Time t_{PHBR} is required for successful relocking of the Boot Block.
8. Sampled but not 100% tested.

1. 25°C, 12.0V V_{PP} .

2. Excludes System-Level Overhead.



290449-18

Figure 18. AC Waveforms for a Write and Erase Operations (WE#-Controlled Writes)

AC CHARACTERISTICS FOR CE #-CONTROLLED WRITE OPERATIONS $V_{CC} = 3.3V \pm 0.3V, 5.0V \pm 10\%$

Versions				28F200BX-L150 28F002BX-L150		Unit
Symbol	Parameter		Notes	Min	Max	
t_{AVAV}	t_{WC}	Write Cycle Time		150		ns
t_{PHEL}	t_{PS}	RP# High Recovery to CE# Going Low		1		μs
t_{WLEL}	t_{WS}	WE# Setup to CE# Going Low		0		ns
t_{PHHEH}	t_{PHS}	RP# V_{HH} Setup to CE# Going High	6, 8	200		ns
t_{VPEH}	t_{VPS}	V_{PP} Setup to CE# Going High	5, 8	200		ns
t_{AVEH}	t_{AS}	Address Setup to CE# Going High	3	95		ns
t_{DVEH}	t_{DS}	Data Setup to CE# Going High	4	100		ns
t_{ELEH}	t_{CP}	CE# Pulse Width		100		ns
t_{EHDX}	t_{DH}	Data Hold from CE# High	4	0		ns
t_{EHAX}	t_{AH}	Address Hold from CE# High	3	10		ns
t_{EWHH}	t_{WH}	WE# Hold from CE# High		10		ns
t_{EHEL}	t_{CPH}	CE# Pulse Width High		50		ns
t_{EHQV1}		Duration of Word/Byte Programming Operation (Boot)	2, 5, 6	6		μs
t_{EHQV2}		Duration of Erase Operation (Boot)	2, 5, 6	0.3		s
t_{EHQV3}		Duration of Erase Operation (Parameter)	2, 5, 6	0.3		s
t_{EHQV4}		Duration of Erase Operation (Main)	2, 5, 6	0.6		s
t_{QWL}	t_{VPH}	V_{PP} Hold from Valid SRD	5, 8	0		ns
t_{QVPH}	t_{PPH}	RP# V_{HH} Hold from Valid SRD	6, 8	0		ns
t_{PHBR}		Boot-Block Relock Delay	7		200	ns

NOTES:

1. Chip-Enable Controlled Writes: Write operations are driven by the valid combination of CE# and WE# in systems where CE# defines the write pulse-width (within a longer WE# timing waveform), all set-up, hold and inactive WE# time should be measured relative to the CE# waveforms.

2, 3, 4, 5, 6, 7, 8: Refer to AC characteristics for WE#-controlled write operations.

9. Read timing characteristics during write and erase operations are the same as during read-only operations. Refer to AC characteristics during read mode.

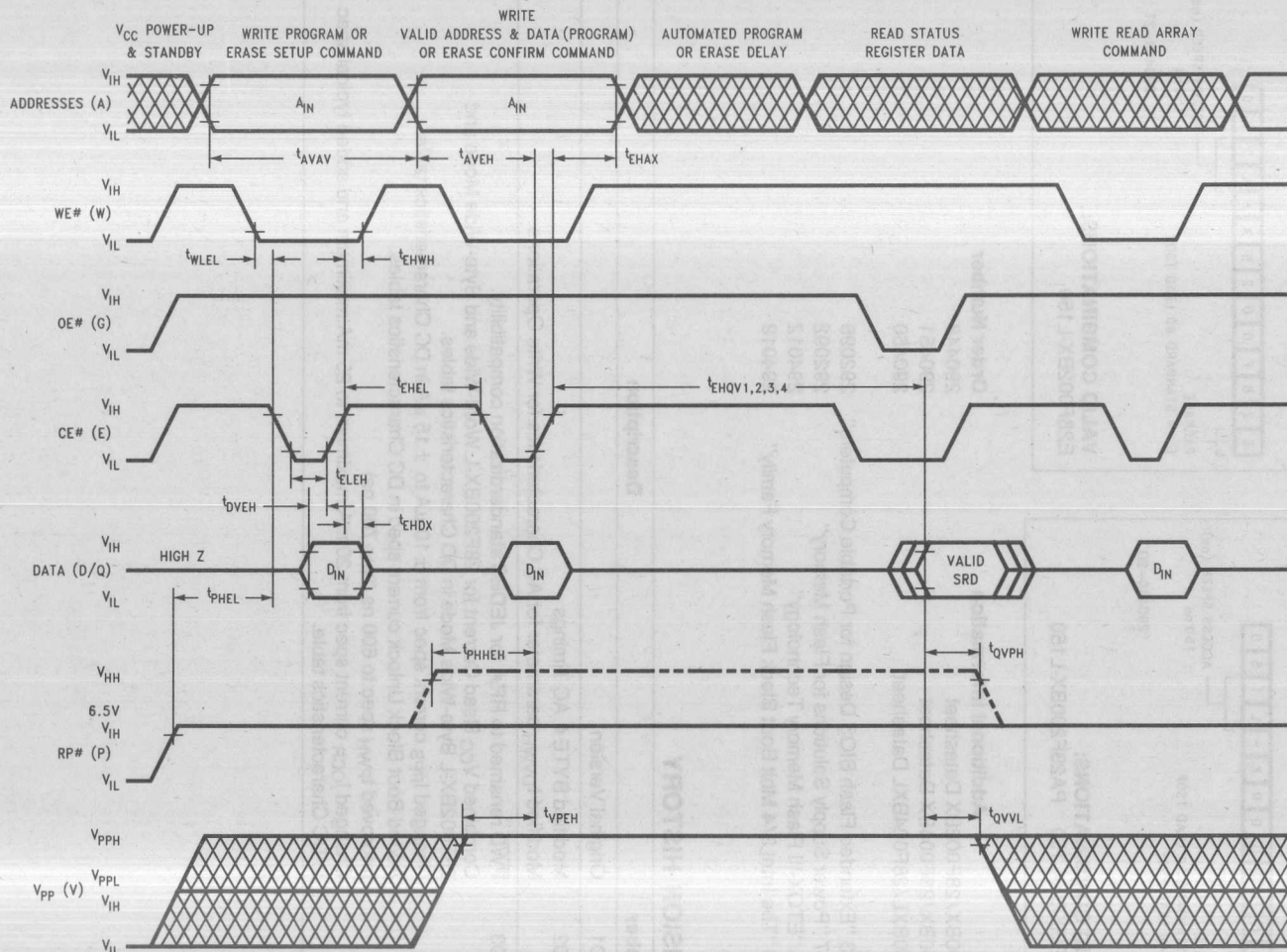


Figure 19. Alternate AC Waveforms for Write and Erase Operations (CE#-Controlled Writes)

PRELIMINARY

ORDERING INFORMATION

E	2	8	F	2	0	0	B	X	-	L	1	5	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---

PACKAGE

E = STANDARD 56 LEAD TSOP
PA = 44 LEAD PSOP

ACCESS SPEED (ns)
150 ns

290449-20

VALID COMBINATIONS:

E28F200BX-L150 PA28F200BX-L150

E	2	8	F	0	0	2	B	X	-	L	1	5	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---

PACKAGE

E = STANDARD 40 LEAD TSOP

ACCESS SPEED (ns)
150 ns

290449-21

VALID COMBINATIONS:

E28F002BX-L150

Additional Information

28F200BX/28F002BX Datasheet
28F400BX/28F004BX Datasheet
28F400BXL/28F004BXL Datasheet

Order Number

290448
290451
290450

AP-363 "Extended Flash BIOS Design for Portable Computers" 292098
AP-357 "Power Supply Solutions for Flash Memory" 292092
ER-28 "ETOX-III Flash Memory Technology" 294012
ER-29 "The Intel 2/4 Mbit Boot Block Flash Memory Family" 294013

REVISION HISTORY

Number	Description
-001	Original Version
-002	Modified BYTE# AC Timings Modified t_{DVWH} parameter for AC Characteristics for Write Operations
-003	\overline{PWD} renamed to RP# for JEDEC standardization compatibility. Combined V_{CC} Read Current for 28F200BX-L Word-Wide and Byte-Wide Mode and 28F002BX-L Byte-Wide Mode in DC Characteristics tables. Changed I_{PPS} current spec from $\pm 10 \mu A$ to $\pm 15 \mu A$ in DC Characteristics table. Added Boot Block Unlock current spec in DC Characteristics tables. Improved t_{PWH} spec to 600 ns (was 700 ns) Changed I_{CCR} current spec from 20 mA maximum to 25 mA maximum and added typical spec to DC Characteristics table.



28F001BX-T/28F001BX-B 1M (128K x 8) CMOS FLASH MEMORY

- **High Integration Blocked Architecture**
 - One 8 KB Boot Block w/Lock Out
 - Two 4 KB Parameter Blocks
 - One 112 KB Main Block
- **100,000 Erase/Program Cycles Per Block**
- **Simplified Program and Erase**
 - Automated Algorithms via On-Chip Write State Machine (WSM)
- **SRAM-Compatible Write Interface**
- **Deep-Powerdown Mode**
 - 0.05 μ A I_{CC} Typical
 - 0.8 μ A I_{pp} Typical
- **12.0V \pm 5% V_{pp}**
- **High-Performance Read**
 - 120 ns Maximum Access Time
 - 5.0V \pm 10% V_{CC}
- **Hardware Data Protection Feature**
 - Erase/Write Lockout during Power Transitions
- **Advanced Packaging, JEDEC Pinouts**
 - 32-Pin PDIP
 - 32-Lead PLCC, TSOP
- **ETOX II Nonvolatile Flash Technology**
 - EPROM-Compatible Process Base
 - High-Volume Manufacturing Experience
- **Extended Temperature Options**

Intel's 28F001BX-B and 28F001BX-T combine the cost-effectiveness of Intel standard flash memory with features that simplify write and allow block erase. These devices aid the system designer by combining the functions of several components into one, making boot block flash an innovative alternative to EPROM and EEPROM or battery-backed static RAM. Many new and existing designs can take advantage of the 28F001BX's integration of blocked architecture, automated electrical reprogramming, and standard processor interface.

The 28F001BX-B and 28F001BX-T are 1,048,576 bit nonvolatile memories organized as 131,072 bytes of 8 bits. They are offered in 32-pin plastic DIP, 32-lead PLCC and 32-lead TSOP packages. Pin assignment conform to JEDEC standards for byte-wide EPROMs. These devices use an integrated command port and state machine for simplified block erasure and byte reprogramming. The 28F001BX-T's block locations provide compatibility with microprocessors and microcontrollers that boot from high memory, such as Intel's MCS-186 family, 80286, i386™, i486™, i860™ and 80960CA. With exactly the same memory segmentation, the 28F001BX-B memory map is tailored for microprocessors and microcontrollers that boot from low memory, such as Intel's MCS-51, MCS-196, 80960KX and 80960SX families. All other features are identical, and unless otherwise noted, the term 28F001BX can refer to either device throughout the remainder of this document.

The boot block section includes a reprogramming write lock out feature to guarantee data integrity. It is designed to contain secure code which will bring up the system minimally and download code to the other locations of the 28F001BX. Intel's 28F001BX employs advanced CMOS circuitry for systems requiring high-performance access speeds, low power consumption, and immunity to noise. Its 120 ns access time provides no-WAIT-state performance for a wide range of microprocessors and microcontrollers. A deep-powerdown mode lowers power consumption to 0.25 μ W typical through V_{CC} , crucial in laptop computer, handheld instrumentation and other low-power applications. The RP# power control input also provides absolute data protection during system powerup or power loss.

Manufactured on Intel's ETOX process base, the 28F001BX builds on years of EPROM experience to yield the highest levels of quality, reliability, and cost-effectiveness.

WE#	WRITE ENABLE: Controls writes to the Command Register and only to the WE# is active low. Addresses and data are latched on the rising edge of the WE# pulse.
Vpp	ERASE/PROGRAM POWER SUPPLY for erasing blocks of the array or programming bytes of each block. Note: With Vpp < Vppmax, memory contents cannot be altered.
VCC	DEVICE POWER SUPPLY (5V \pm 10%)
GROUND	

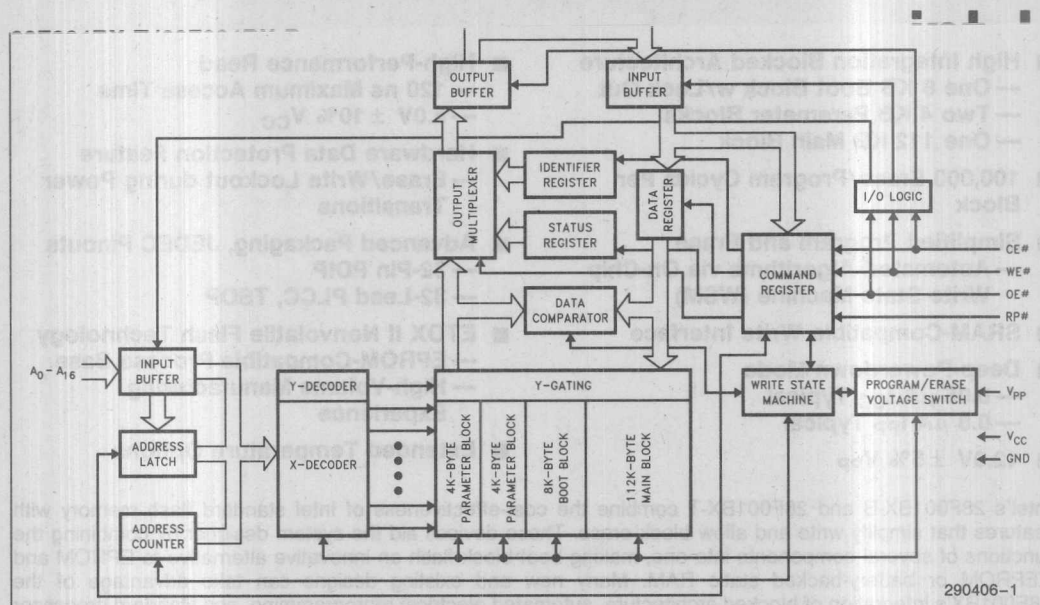


Figure 1. 28F001BX Block Diagram

Table 1. Pin Description

Symbol	Type	Name and Function
A ₀ -A ₁₆	INPUT	ADDRESS INPUTS for memory addresses. Addresses are internally latched during a write cycle.
DQ ₀ -DQ ₇	INPUT/ OUTPUT	DATA INPUTS/OUTPUTS: Inputs data and commands during memory write cycles; outputs data during memory, Status Register and Identifier read cycles. The data pins are active high and float to tri-state off when the chip is deselected or the outputs are disabled. Data is internally latched during a write cycle.
CE #	INPUT	CHIP ENABLE: Activates the device's control logic, input buffers, decoders and sense amplifiers. CE # is active low; CE # high deselects the memory device and reduces power consumption to standby levels.
RP #	INPUT	POWERDOWN: Puts the device in deep powerdown mode. RP # is active low; RP # high gates normal operation. RP # = V _{HH} allows programming of the boot block. RP # also locks out erase or write operations when active low, providing data protection during power transitions. RP # active resets internal automation. Exit from deep powerdown sets device to Read Array mode.
OE #	INPUT	OUTPUT ENABLE: Gates the device's outputs through the data buffers during a read cycle. OE # is active low. OE # = V _{HH} (pulsed) allows programming of the boot block.
WE #	INPUT	WRITE ENABLE: Controls writes to the Command Register and array blocks. WE # is active low. Addresses and data are latched on the rising edge of the WE # pulse.
V _{PP}		ERASE/PROGRAM POWER SUPPLY for erasing blocks of the array or programming bytes of each block. Note: With V _{PP} < V _{PP} L max, memory contents cannot be altered.
V _{CC}		DEVICE POWER SUPPLY: (5V ± 10%)
GND		GROUND

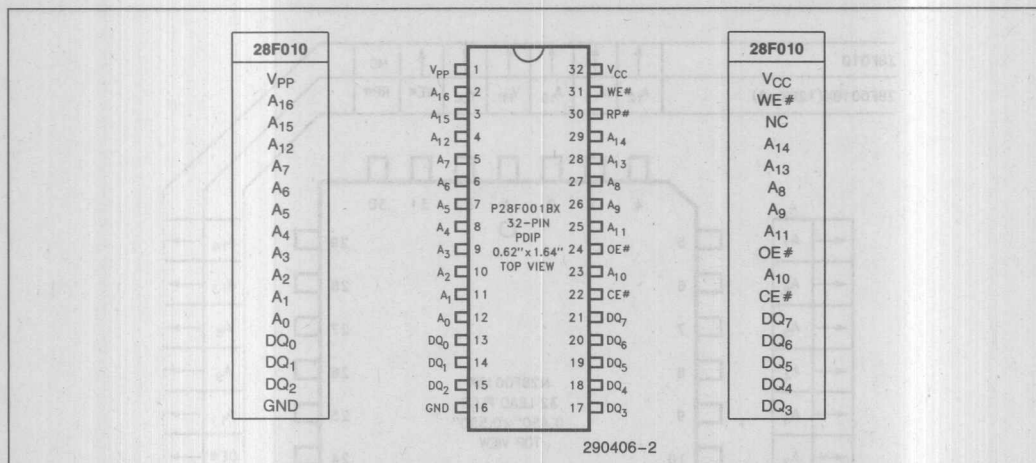


Figure 2. DIP Pin Configuration

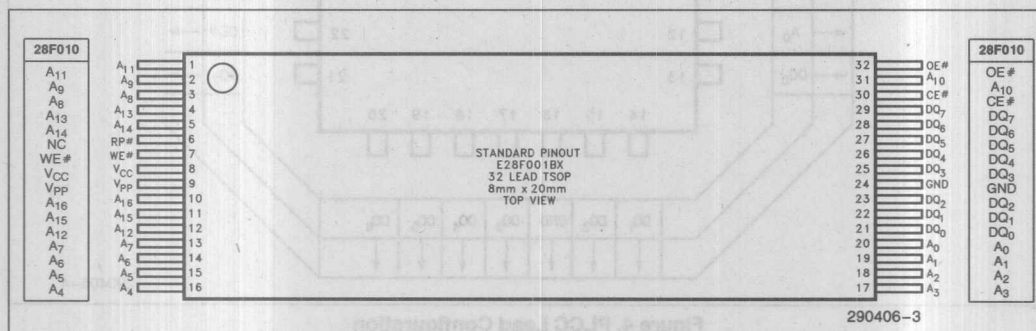


Figure 3. TSOP Lead Configuration

throughout the life cycle of a design. During the early stages of a system's life, flash memory reduces development and testing time, allowing the system designer to modify in-system software electrically versus manual removal of components. During production, flash memory provides flexible firmware for just-in-time configuration, reducing system inventory and eliminating unnecessary handling and less valuable stocked components. Later in the life cycle, when software updates or "code bugs" are often unavoidable and costly, flash memory reduces update costs by allowing the manufacturer to send floppy updates via a technical Atlanta-based remote updates over a communication link and possible at speeds up to 5000 baud due to flash memory's fast programming time.

APPLICATIONS

The 28F001BX flash "boot block" memory augments the non-volatile in-system electrical erasure and reprogramming ability of Intel's standard flash memory by offering fast, non-volatile erase blocks and intelligent a state machine to control erase and program functions. The specialized blocking architecture and automated programming of the 28F001BX provide a full-function, non-volatile flash memory ideal for a wide range of applications, including PC boot block memory, minimum-size embedded program memory and parametric data storage. The 28F001BX provides the safety of a hardware-protected 4-Kbit boot block with the flexibility of three separately reprogrammable blocks (two 4-Kbit parameter blocks and one 12-Kbit code block) into one versatile, cost-effective flash memory. Additionally, reprogramming one block does not affect code stored in another block, ensuring data integrity.

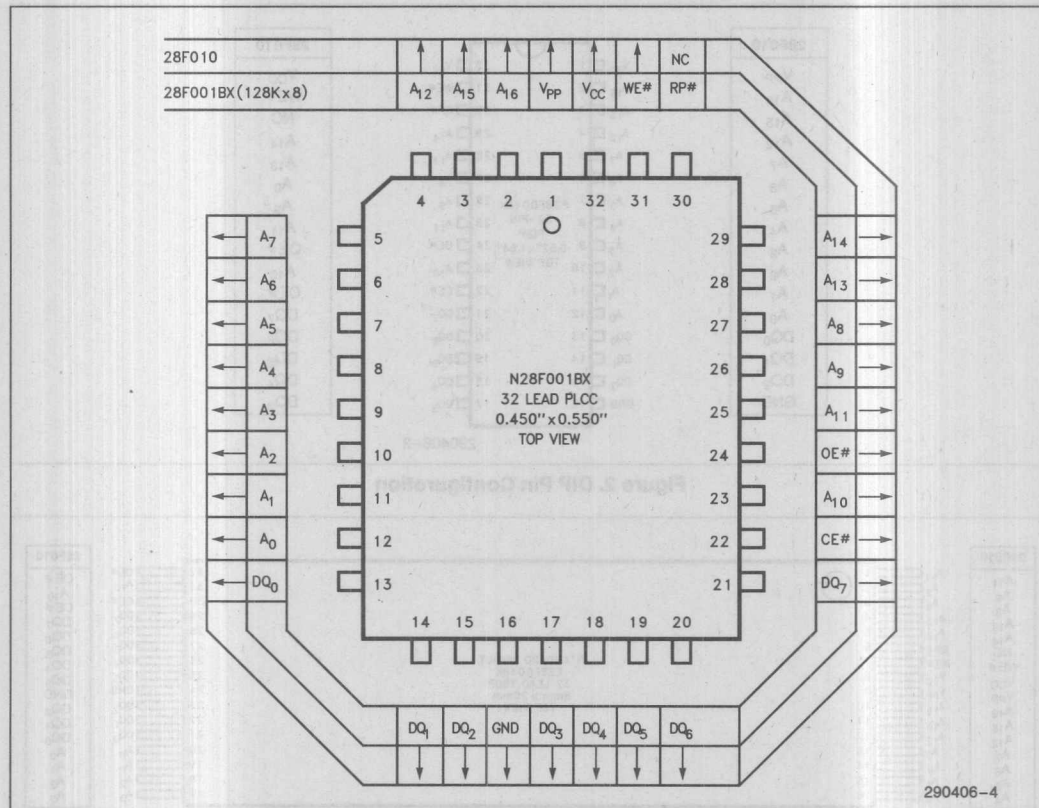


Figure 4. PLCC Lead Configuration

APPLICATIONS

The 28F001BX flash 'boot block' memory augments the non-volatility, in-system electrical erasure and reprogrammability of Intel's standard flash memory by offering four separately erasable blocks and integrating a state machine to control erase and program functions. The specialized blocking architecture and automated programming of the 28F001BX provide a full-function, non-volatile flash memory ideal for a wide range of applications, including PC boot/BIOS memory, minimum-chip embedded program memory and parametric data storage. The 28F001BX combines the safety of a hardware-protected 8-KByte boot block with the flexibility of three separately reprogrammable blocks (two 4-KByte parameter blocks and one 112-KByte code block) into one versatile, cost-effective flash memory. Additionally, reprogramming one block does not affect code stored in another block, ensuring data integrity.

The flexibility of flash memory reduces costs throughout the life cycle of a design. During the early stages of a system's life, flash memory reduces prototype development and testing time, allowing the system designer to modify in-system software electrically versus manual removal of components. During production, flash memory provides flexible firmware for just-in-time configuration, reducing system inventory and eliminating unnecessary handling and less reliable socketed connections. Late in the life cycle, when software updates or code "bugs" are often unpredictable and costly, flash memory reduces update costs by allowing the manufacturers to send floppy updates versus a technician. Alternatively, remote updates over a communication link are possible at speeds up to 9600 baud due to flash memory's fast programming time.

sonal computer, are ideal applications for the 28F001BX. The internal state machine provides SRAM-like timings for program and erasure, using the Command and Status Registers. The blocking scheme allows BIOS update in the main and parameter blocks, while still providing recovery code in the boot block in the unlikely event a power failure occurs during an update, or where BIOS code is corrupted. Parameter blocks also provide convenient configuration storage, backing up SRAM and battery configurations. EISA systems, for example, can store hardware configurations in a flash parameter block, reducing system SRAM.

Laptop BIOSs are becoming increasingly complex with the addition of power management software and extended system setup screens. BIOS code complexity increases the potential for code updates after the sale, but the compactness of laptop designs makes hardware updates very costly. Boot block flash memory provides an inexpensive update solution for laptops, while reducing laptop obsolescence. For portable PCs and hand-held equipment, the deep powerdown mode dramatically lowers sys-

eration or sleep modes.

The 28F001BX gives the embedded system designer several desired features. The internal state machine reduces the size of external code dedicated to the erase and program algorithms, as well as freeing the microcontroller or microprocessor to respond to other system requests during program and erasure. The four blocks allow logical segmentation of the entire embedded software: the 8-KByte block for the boot code, the 112-KByte block for the main program code and the two 4-KByte blocks for updatable parametric data storage, diagnostic messages and data, or extensions of either the boot code or program code. The boot block is hardware protected against unauthorized write or erase of its vital code in the field. Further, the powerdown mode also locks out erase or write operations, providing absolute data protection during system powerup or power loss. This hardware protection provides obvious advantages for safety related applications such as transportation, military, and medical. The 28F001BX is well suited for minimum-chip embedded applications ranging from communications to automotive.

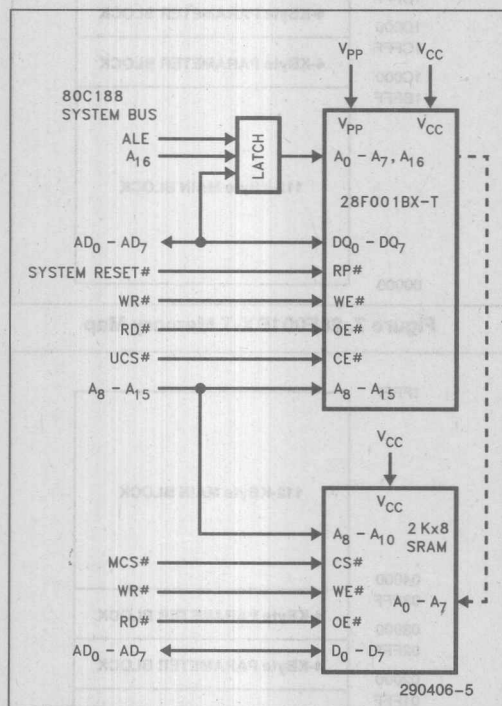


Figure 5. 28F001BX-T in a 80C188 System

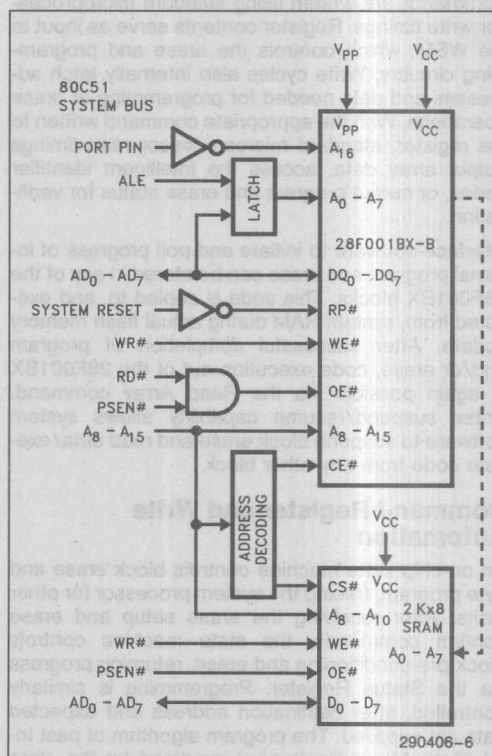


Figure 6. 28F001BX-B in a 80C51 System

PRINCIPLES OF OPERATION

The 28F001BX introduces on-chip write automation to manage write and erase functions. The write state machine allows for: 100% TTL-level control inputs; fixed power supplies during erasure and programming; minimal processor overhead with RAM-like write timings, and maximum EPROM compatibility.

After initial device powerup, or after return from deep powerdown mode (see Bus Operations), the 28F001BX functions as a read-only memory. Manipulation of external memory-control pins yield standard EPROM read, standby, output disable or Intelligent Identifier operations. Both Status Register and Intelligent Identifiers can be accessed through the Command Register when $V_{PP} = V_{PPL}$.

This same subset of operations is also available when high voltage is applied to the V_{PP} pin. In addition, high voltage on V_{PP} enables successful erasure and programming of the device. All functions associated with altering memory contents—program, erase, status, and intelligent identifier—are accessed via the Command Register and verified through the Status Register.

Commands are written using standard microprocessor write timings. Register contents serve as input to the WSM, which controls the erase and programming circuitry. Write cycles also internally latch addresses and data needed for programming or erase operations. With the appropriate command written to the register, standard microprocessor read timings output array data, access the intelligent identifier codes, or output program and erase status for verification.

Interface software to initiate and poll progress of internal program and erase can be stored in any of the 28F001BX blocks. This code is copied to, and executed from, system RAM during actual flash memory update. After successful completion of program and/or erase, code execution out of the 28F001BX is again possible via the Read Array command. Erase suspend/resume capability allows system software to suspend block erase and read data/execute code from any other block.

Command Register and Write Automation

An on-chip state machine controls block erase and byte program, freeing the system processor for other tasks. After receiving the erase setup and erase confirm commands, the state machine controls block pre-conditioning and erase, returning progress via the Status Register. Programming is similarly controlled, after destination address and expected data are supplied. The program algorithm of past Intel Flash Memories is now regulated by the state machine, including program pulse repetition where required and internal verification and margining of data.

Data Protection

Depending on the application, the system designer may choose to make the V_{PP} power supply switchable (available only when memory updates are required) or hardwired to V_{PPH} . When $V_{PP} = V_{PPL}$, memory contents cannot be altered. The 28F001BX Command Register architecture provides protection from unwanted program or erase operations even when high voltage is applied to V_{PP} . Additionally, all functions are disabled whenever V_{CC} is below the write lockout voltage V_{LKO} , or when $RP\#$ is at V_{IL} . The 28F001BX accommodates either design practice and encourages optimization of the processor-memory interface.

The two-step program/erase write sequence to the Command Register provides additional software write protection.

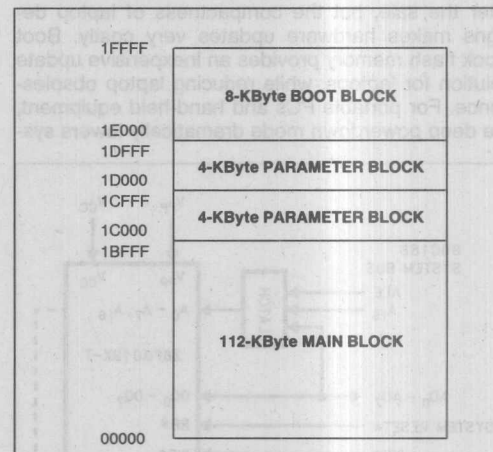


Figure 7. 28F001BX-T Memory Map

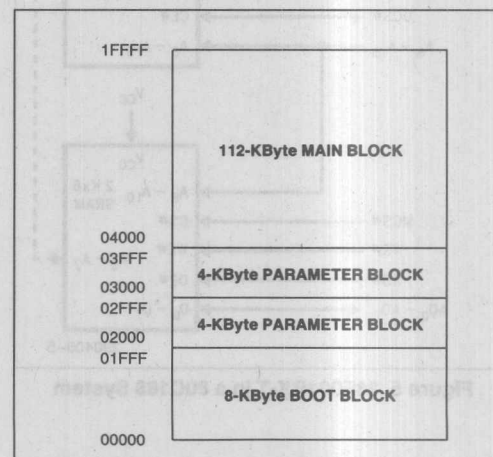


Figure 8. 28F001BX-B Memory Map

BUS OPERATION

Flash memory reads, erases and writes in-system via the local CPU. All bus cycles to or from the flash memory conform to standard microprocessor bus cycles.

Read

The 28F001BX has three read modes. The memory can be read from any of its blocks, and information can be read from the Intelligent Identifier or the Status Register. V_{PP} can be at either V_{PPL} or V_{PPH} .

The first task is to write the appropriate read mode command to the Command Register (array, Intelligent Identifier, or Status Register). The 28F001BX automatically resets to Read Array mode upon initial device powerup or after exit from deep powerdown. The 28F001BX has four control pins, two of which must be logically active to obtain data at the outputs. Chip Enable ($CE\#$) is the device selection control, and when active enables the selected memory device. Output Enable ($OE\#$) is the data input/output (DQ_0 – DQ_7) direction control, and when active drives data from the selected memory onto the I/O bus. $RP\#$ and $WE\#$ must also be at V_{IH} . Figure 12 illustrates read bus cycle waveforms.

Output Disable

With $OE\#$ at a logic-high level (V_{IH}), the device outputs are disabled. Output pins (DQ_0 – DQ_7) are placed in a high-impedance state.

Standby

$CE\#$ at a logic-high level (V_{IH}) places the 28F001BX in standby mode. Standby operation disables much of the 28F001BX's circuitry and substantially reduces device power consumption. The outputs (DQ_0 – DQ_7) are placed in a high-impedance state independent of the status of $OE\#$. If the 28F001BX is deselected during erase or program, the device will continue functioning and consuming normal active power until the operation is completed.

Deep Power-Down

The 28F001BX offers a $0.25\ \mu W$ V_{CC} power-down feature, entered when $RP\#$ is at V_{IL} . During read modes, $RP\#$ low deselects the memory, places output drivers in a high-impedance state and turns off all internal circuits. The 28F001BX requires time t_{PHQV} (see AC Characteristics-Read Only Operations) after return from power-down until initial memory access outputs are valid. After this wakeup interval, normal operation is restored. The Command Register is reset to Read Array, and the Status Register is cleared to value 80H, upon return to normal operation.

During erase or program modes, $RP\#$ low will abort either operation. Memory contents of the block being altered are no longer valid as the data will be partially programmed or erased. Time t_{PHWL} after $RP\#$ goes to logic-high (V_{IH}) is required before another command can be written.

Table 2. 28F001BX Bus Operations

Mode	Notes	$RP\#$	$CE\#$	$OE\#$	$WE\#$	A_9	A_0	V_{PP}	DQ_0 –7
Read	1, 2, 3	V_{IH}	V_{IL}	V_{IL}	V_{IH}	X	X	X	D_{OUT}
Output Disable		V_{IH}	V_{IL}	V_{IH}	V_{IH}	X	X	X	High Z
Standby		V_{IH}	V_{IH}	X	X	X	X	X	High Z
Deep Power Down		V_{IL}	X	X	X	X	X	X	High Z
Intelligent Identifier (Mfr)	4	V_{IH}	V_{IL}	V_{IL}	V_{IH}	V_{ID}	V_{IL}	X	89H
Intelligent Identifier (Device)	4, 5	V_{IH}	V_{IL}	V_{IL}	V_{IH}	V_{ID}	V_{IH}	X	94H, 95H
Write	6, 7, 8	V_{IH}	V_{IL}	V_{IH}	V_{IL}	X	X	X	D_{IN}

NOTES:

- Refer to DC Characteristics. When $V_{PP} = V_{PPL}$, memory contents can be read but not programmed or erased.
- X can be V_{IL} or V_{IH} for control pins and addresses, and V_{PPL} or V_{PPH} for V_{PP} .
- See DC Characteristics for V_{PPL} , V_{PPH} , V_{HH} and V_{ID} voltages.
- Manufacturer and device codes may also be accessed via a Command Register write sequence. Refer to Table 3. A_1 – A_8 , A_{10} – $A_{16} = V_{IL}$.
- Device ID = 94H for the 28F001BX-T and 95H for the 28F001BX-B.
- Command writes involving block erase or byte program are successfully executed only when $V_{PP} = V_{PPH}$.
- Refer to Table 3 for valid D_{IN} during a write operation.
- Program or erase the boot block by holding $RP\#$ at V_{HH} or toggling $OE\#$ to V_{HH} . See AC Waveforms for program/erase operations.

The use of RP# during system reset is important with automated write/erase devices. When the system comes out of reset it expects to read from the flash memory. Automated flash memories provide status information when accessed during write/erase modes. If a CPU reset occurs with no flash memory reset, proper CPU initialization would not occur because the flash memory would be providing the status information instead of array data. Intel's Flash Memories allow proper CPU initialization following a system reset through the use of the RP# input. In this application RP# is controlled by the same RESET# signal that resets the system CPU.

Intelligent Identifier Operation

The Intelligent Identifier operation outputs the manufacturer code, 89H; and the device code, 94H for the 28F001BX-T and 95H for the 28F001BX-B. Programming equipment or the system CPU can then automatically match the device with its proper erase and programming algorithms.

PROGRAMMING EQUIPMENT

CE# and OE# at a logic low level (V_{IL}), with A_9 at high voltage V_{ID} (see DC Characteristics) activates this operation. Data read from locations 00000H and 00001H represent the manufacturer's code and the device code respectively.

IN-SYSTEM PROGRAMMING

The manufacturer- and device-codes can also be read via the Command Register. Following a write of 90H to the Command Register, a read from address location 00000H outputs the manufacturer code (89H). A read from address 00001H outputs the device code (94H for the 28F001BX-T and 95H for the 28F001BX-B). It is not necessary to have high voltage applied to V_{pp} to read the Intelligent Identifiers from the Command Register.

Write

Writes to the Command Register allow read of device data and Intelligent Identifiers. They also control inspection and clearing of the Status Register. Additionally, when $V_{pp} = V_{ppH}$, the Command Register controls device erasure and programming. The contents of the register serve as input to the internal state machine.

The Command Register itself does not occupy an addressable memory location. The register is a latch used to store the command and address and data information needed to execute the command. Erase

Setup and Erase Confirm commands require both appropriate command data and an address within the block to be erased. The Program Setup Command requires both appropriate command data and the address of the location to be programmed, while the Program command consists of the data to be written and the address of the location to be programmed.

The Command Register is written by bringing WE# to a logic-low level (V_{IL}) while CE# is low. Addresses and data are latched on the rising edge of WE#. Standard microprocessor write timings are used.

Refer to AC Write Characteristics and the AC Waveform for Write Operations, Figure 13, for specific timing parameters.

COMMAND DEFINITIONS

When V_{pPL} is applied to the V_{pp} pin, read operations from the Status Register, intelligent identifiers, or array blocks are enabled. Placing V_{ppH} on V_{pp} enables successful program and erase operations as well.

Device operations are selected by writing specific commands into the Command Register. Table 3 defines these 28F001BX commands.

Read Array Command

Upon initial device powerup and after exit from deep-powerdown mode, the 28F001BX defaults to Read Array mode. This operation is also initiated by writing FFH into the Command Register. Microprocessor read cycles retrieve array data. The device remains enabled for reads until the Command Register contents are altered. Once the internal write state machine has started an erase or program operation, the device will not recognize the Read Array command, until the WSM has completed its operation. The Read Array command is functional when $V_{pp} = V_{pPL}$ or V_{ppH} .

Intelligent Identifier Command for In-System Programming

The 28F001BX contains an Intelligent Identifier operation to supplement traditional PROM-programming methodology. The operation is initiated by writing 90H into the Command Register. Following the command write, a read cycle from address 00000H retrieves the manufacturer code of 89H. A read cycle from address 00001H returns the device code of 94H (28F001BX-T) or 95H (28F001BX-B). To terminate the operation, it is necessary to write another valid command into the register. Like the Read Array command, the Intelligent Identifier command is functional when $V_{pp} = V_{pPL}$ or V_{ppH} .

Table 3. 28F001BX Command Definitions

Command	Bus Cycles Req'd	Notes	First Bus Cycle			Second Bus Cycle		
			Operation	Address	Data	Operation	Address	Data
Read Array/Reset	1	1	Write	X	FFH			
Intelligent Identifier	3	2, 3, 4	Write	X	90H	Read	IA	IID
Read Status Register	2	3	Write	X	70H	Read	X	SRD
Clear Status Register	1		Write	X	50H			
Erase Setup/Erase Confirm	2	2	Write	BA	20H	Write	BA	D0H
Erase Suspend/Erase Resume	2		Write	X	B0H	Write	X	D0H
Program Setup/Program	2	2, 3	Write	PA	40H	Write	PA	PD

NOTES:

- Bus operations are defined in Table 2.
- IA = Identifier Address: 00H for manufacturer code, 01H for device code.
BA = Address within the block being erased.
PA = Address of memory location to be programmed.
- SRD = Data read from Status Register. See Table 4 for a description of the Status Register bits.
PD = Data to be programmed at location PA. Data is latched on the rising edge of WE#.
IID = Data read from Intelligent Identifiers.
- Following the Intelligent Identifier command, two read operations access manufacture and device codes.
- Commands other than those shown above are reserved by Intel for future device implementations and should not be used.

Read Status Register Command

The 28F001BX contains a Status Register which may be read to determine when a program or erase operation is complete, and whether that operation completed successfully. The Status Register may be read at any time by writing the Read Status Register command (70H) to the Command Register. After writing this command, all subsequent read operations output data from the Status Register, until another valid command is written to the Command Register. The contents of the Status Register are latched on the falling edge of OE# or CE#, whichever occurs last in the read cycle. OE# or CE# must be toggled to VIH before further reads to update the Status Register latch. The Read Status Register command functions when VPP = VPLL or VPPH.

Clear Status Register Command

The Erase Status and Program Status bits are set to "1" by the Write State Machine and can only be

reset by the Clear Status Register command. These bits indicate various failure conditions (see Table 4). By allowing system software to control the resetting of these bits, several operations may be performed (such as cumulatively programming several bytes or erasing multiple blocks in sequence). The Status Register may then be polled to determine if an error occurred during that series. This adds flexibility to the way the device may be used.

Additionally, the Vpp Status bit (SR.3), when set to "1", MUST be reset by system software before further byte programs or block erases are attempted. To clear the Status Register, the Clear Status Register command (50H) is written to the Command Register. The Clear Status Register command is functional when VPP = VPLL or VPPH.

Table 4. 28F001BX Status Register Definitions

WSMS	ESS	ES	PS	VPPS	R	R	R
7	6	5	4	3	2	1	0
SR.7 = WRITE STATE MACHINE STATUS 1 = Ready 0 = Busy				NOTES: The Write State Machine Status Bit must first be checked to determine program or erase completion, before the Program or Erase Status bits are checked for success. If the Program AND Erase Status bits are set to "1s" during an erase attempt, an improper command sequence was entered. Attempt the operation again. If V _{pp} low status is detected, the Status Register must be cleared before another program or erase operation is attempted. The V _{pp} Status bit, unlike an A/D converter, does not provide continuous indication of V _{pp} level. The WSM interrogates the V _{pp} level only after the program or erase command sequences have been entered and informs the system if V _{pp} has not been switched on. The V _{pp} Status bit is not guaranteed to report accurate feedback between V _{ppL} and V _{ppH} .			
SR.6 = ERASE SUSPEND STATUS 1 = Erase Suspended 0 = Erase In Progress/Completed							
SR.5 = ERASE STATUS 1 = Error in Block Erasure 0 = Successful Block Erase							
SR.4 = PROGRAM STATUS 1 = Error in Byte Program 0 = Successful Byte Program							
SR.3 = V _{pp} STATUS 1 = V _{pp} Low Detect; Operation Abort 0 = V _{pp} OK							
SR.2–SR.0 = RESERVED FOR FUTURE ENHANCEMENTS These bits are reserved for future use and should be masked out when polling the Status Register.							

Erase Setup/Erase Confirm Commands

Erase is executed one block at a time, initiated by a two-cycle command sequence. An Erase Setup command (20H) is first written to the Command Register, followed by the Erase Confirm command (D0H). These commands require both appropriate command data and an address within the block to be erased. Block preconditioning, erase and verify are all handled internally by the Write State Machine, invisible to the system. After receiving the two-command erase sequence, the 28F001BX automatically outputs Status Register data when read (see Figure 10; Block Erase Flowchart). The CPU can detect the completion of the erase event by checking the WSM Status bit of the Status Register (SR.7).

When the Status Register indicates that erase is complete, the Erase Status bit should be checked. If erase error is detected, the Status Register should be cleared. The Command Register remains in Read Status Register Mode until further commands are issued to it.

This two-step sequence of set-up followed by execution ensures that memory contents are not accidentally erased. Also, block erasure can only occur when $V_{pp} = V_{ppH}$. In the absence of this high voltage, memory contents are protected against erasure. If block erase is attempted while $V_{pp} = V_{ppL}$,

the V_{pp} Status bit will be set to "1". Erase attempts while $V_{ppL} < V_{pp} < V_{ppH}$ produce spurious results and should not be attempted.

Erase Suspend/Erase Resume Commands

The Erase Suspend Command allows erase sequence interruption in order to read data from another block of memory. Once the erase sequence is started, writing the Erase Suspend command (B0H) to the Command Register requests that the WSM suspend the erase sequence at a predetermined point in the erase algorithm. The 28F001BX continues to output Status Register data when read, after the Erase Suspend command is written to it. Polling the WSM Status and Erase Suspend Status bits will determine when the erase operation has been suspended (both will be set to "1s").

At this point, a Read Array command can be written to the Command Register to read data from blocks **other than that which is suspended**. The only other valid commands at this time are Read Status Register (70H) and Erase Resume (D0H), at which time the WSM will continue with the erase sequence. The Erase Suspend Status and WSM Status bits of the Status Register will be cleared. After the Erase Resume command is written to it, the 28F001BX automatically outputs Status Register data when read (see Figure 11; Erase Suspend/Resume Flowchart).

Program Setup/Program Commands

Programming is executed by a two-write sequence. The program Setup command (40H) is written to the Command Register, followed by a second write specifying the address and data (latched on the rising edge of WE#) to be programmed. The WSM then takes over, controlling the program and verify algorithms internally. After the two-command program sequence is written to it, the 28F001BX automatically outputs Status Register data when read (see figure 9; Byte Program Flowchart). The CPU can detect the completion of the program event by analyzing the WSM Status bit of the Status Register. Only the Read Status Register command is valid while programming is active.

When the Status Register indicates that programming is complete, the Program Status bit should be checked. If program error is detected, the Status Register should be cleared. The internal WSM verify only detects errors for "1s" that do not successfully program to "0s". The Command Register remains in Read Status Register mode until further commands are issued to it. If byte program is attempted while $V_{PP} = V_{PPL}$, the V_{PP} Status bit will be set to "1". Program attempts while $V_{PPL} < V_{PP} < V_{PPH}$ produce spurious results and should not be attempted.

EXTENDED ERASE/PROGRAM CYCLING

EEPROM cycling failures have always concerned users. The high electrical field required by thin oxide EEPROMs for tunneling can literally tear apart the oxide at defect regions. To combat this, some suppliers have implemented redundancy schemes, reducing cycling failures to insignificant levels. However, redundancy requires that cell size be doubled; an expensive solution.

Intel has designed extended cycling capability into its ETOX flash memory technology. Resulting improvements in cycling reliability come without increasing memory cell size or complexity. First, an advanced tunnel oxide increases the charge carrying ability ten-fold. Second, the oxide area per cell subjected to the tunneling electrical field is one-tenth that of common EEPROMs, minimizing the probability of oxide defects in the region. Finally, the peak electric field during erasure is approximately 2 Mv/cm lower than EEPROM. The lower electric field greatly reduces oxide stress and the probability of failure; increasing time to wearout by a factor of 100,000,000.

The 28F001BX-B and 28F001BX-T are capable of 100,000 program/erase cycles on each of the two parameter blocks, main block and boot block.

ON-CHIP PROGRAMMING ALGORITHM

The 28F001BX integrates the Quick Pulse programming algorithm of prior Intel Flash Memory devices on-chip, using the Command Register, Status Register and Write State Machine (WSM). On-chip integration dramatically simplifies system software and provides processor-like interface timings to the Command and Status Registers. WSM operation, internal program verify and V_{PP} high voltage presence are monitored and reported via appropriate Status Register bits. Figure 9 shows a system software flowchart for device programming. The entire sequence is performed with V_{PP} at V_{PPH} . Program abort occurs when RP# transitions to V_{IL} , or V_{PP} drops to V_{PPL} . Although the WSM is halted, byte data is partially programmed at the location where programming was aborted. Block erasure or a repeat of byte programming will initialize this data to a known value.

ON-CHIP ERASE ALGORITHM

As above, the Quick Erase algorithm of prior Intel Flash Memory devices is now implemented internally, including all preconditioning of block data. WSM operation, erase success and V_{PP} high voltage presence are monitored and reported through the Status Register. Additionally, if a command other than Erase Confirm is written to the device after Erase Setup has been written, both the Erase Status and Program Status bits will be set to "1". When issuing the Erase Setup and Erase Confirm commands, they should be written to an address within the address range of the block to be erased. Figure 10 shows a system software flowchart for block erase.

Erase typically takes 1–4 seconds per block. The Erase Suspend/Erase Resume command sequence allows interrupt of this erase operation to read data **from a block other than that in which erase is being performed**. A system software flowchart is shown in Figure 11.

The entire sequence is performed with V_{PP} at V_{PPH} . Abort occurs when RP# transitions to V_{IL} or V_{PP} falls to V_{PPL} , while erase is in progress. Block data is partially erased by this operation, and a repeat of erase is required to obtain a fully erased block.

ERASE

The boot block is intended to contain secure code which will minimally bring up a system and control programming and erase of other blocks of the device, if needed. Therefore, additional "lockout" protection is provided to guarantee data integrity. Boot block program and erase operations are enabled through high voltage V_{HH} on either RP# or OE#, and the normal program and erase command sequences are used. Reference the AC Waveforms for Program/Erase.

If boot block program or erase is attempted while RP# is at V_{IH} , either the Program Status or Erase Status bit will be set to "1", reflective of the opera-

Program/erase attempts while $V_{IH} < RP\# < V_{HH}$ produce spurious results and should not be attempted.

In-System Operation

For on-board programming, the RP# pin is the most convenient means of altering the boot block. Before issuing Program or Erase confirms commands, RP# must transition to V_{HH} . Hold RP# at this high voltage throughout the program or erase interval (until after Status Register confirm of successful completion). At this time, it can return to V_{IH} or V_{IL} .

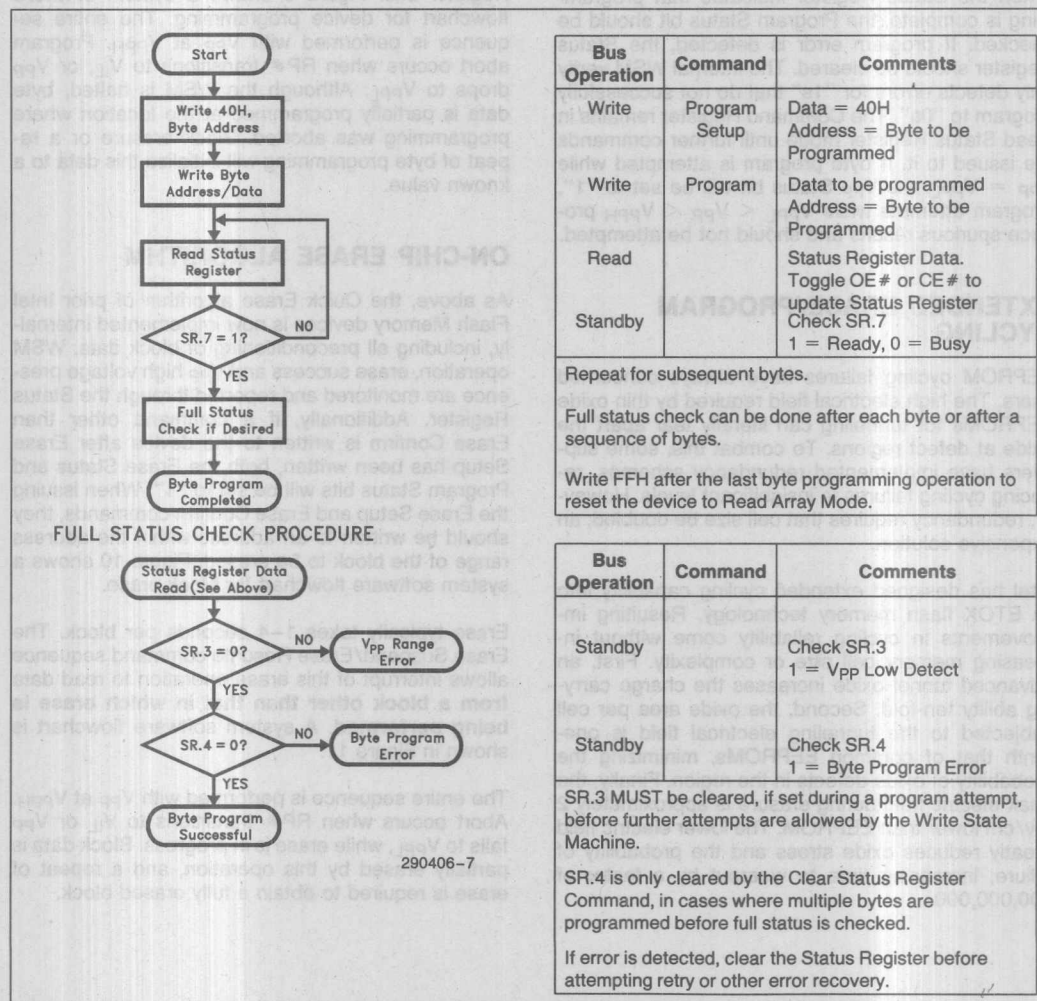


Figure 9. 28F001BX Byte Programming Flowchart

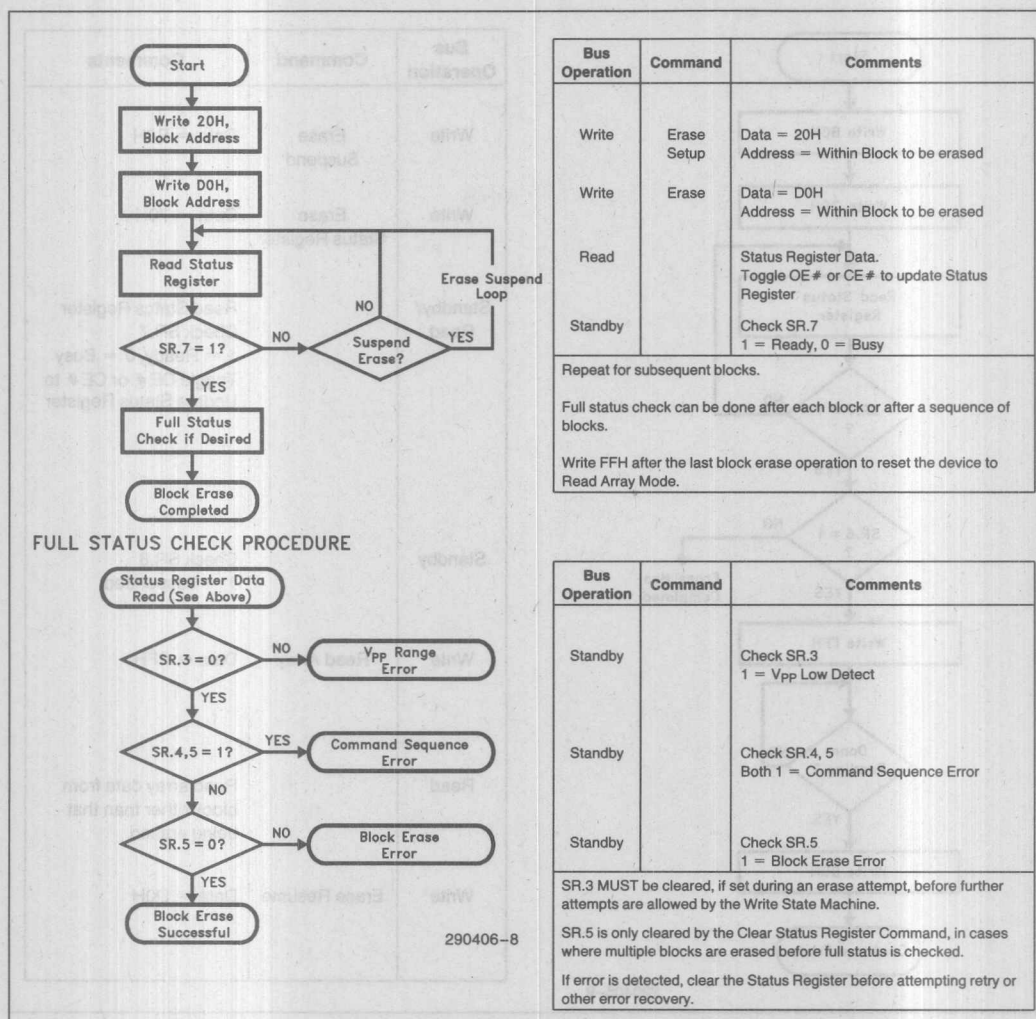


Figure 10. 28F001BX Block Erase Flowchart

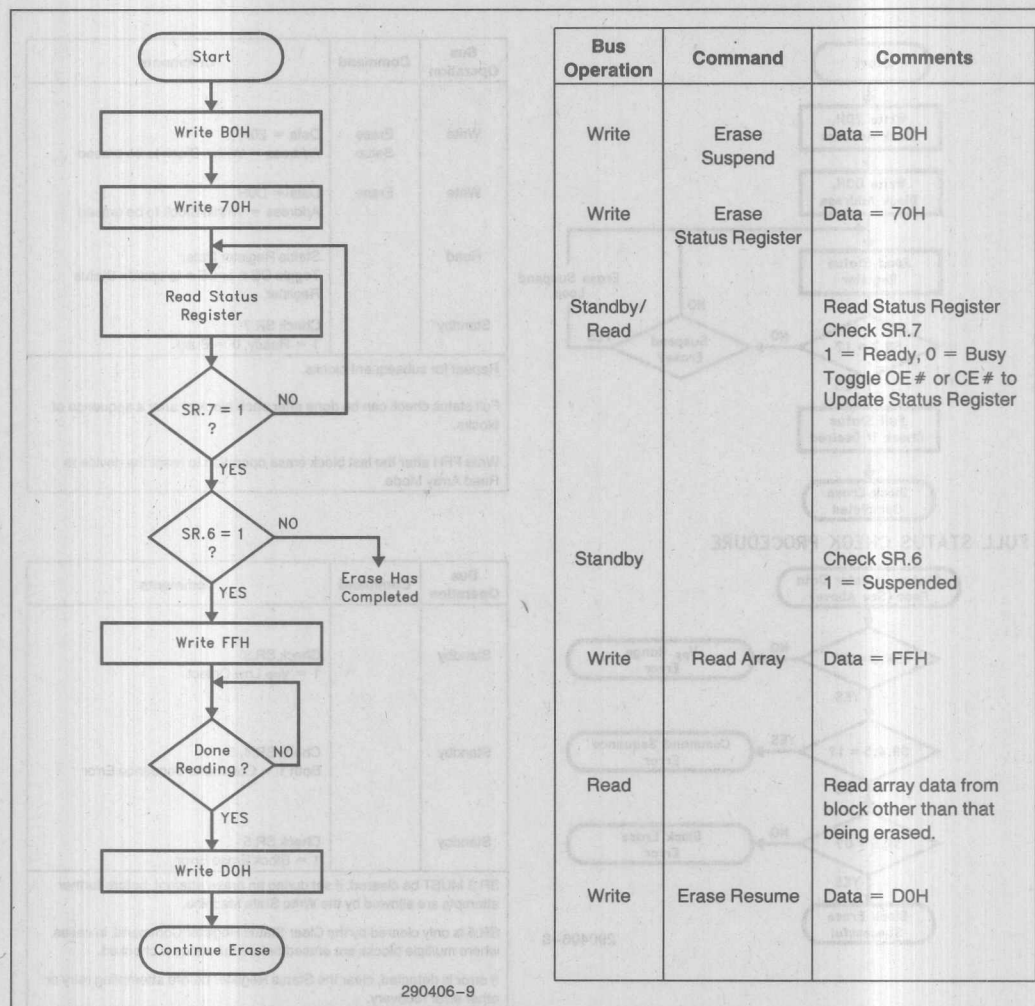


Figure 11. 28F001BX Erase Suspend/Resume Flowchart

Programming Equipment

For PROM programming equipment that cannot bring RP# to high voltage, OE# provides an alternate boot block access mechanism. OE# must transition to V_{HH} a minimum of 480 ns before the initial program/erase setup command and held at V_{HH} at least 480 ns after program or erase confirm commands are issued to the device. After this interval, OE# can return to normal TTL levels.

DESIGN CONSIDERATIONS**Three-Line Output Control**

Flash memories are often used in larger memory arrays. Intel provides three control inputs to accommo-

date multiple memory connections. Three-line control provides for:

- lowest possible memory power dissipation
- complete assurance that data bus contention will not occur

To efficiently use these control inputs, an address decoder should enable CE#, while OE# should be connected to all memory devices and the system's READ# control line. This assures that only selected memory devices have active outputs while deselected memory devices are in Standby Mode. RP# should be connected to the system POWERGOOD signal to prevent unintended writes during system power transitions. POWERGOOD should also toggle during system reset.

Flash memory power switching characteristics require careful device coupling. System designers are interested in 3 supply current issues; standby current levels (I_{SB}), active current levels (I_{CC}) and transient peaks produced by falling and rising edges of $CE\#$. Transient current magnitudes depend on the device outputs' capacitive and inductive loading. Two-line control and proper decoupling capacitor selection will suppress transient voltage peaks. Each device should have a $0.1\ \mu\text{F}$ ceramic capacitor connected between its V_{CC} and GND, and between its V_{PP} and GND. These high frequency, low inherent-inductance capacitors should be placed as close as possible to the device. Additionally, for every 8 devices, a $4.7\ \mu\text{F}$ electrolytic capacitor should be placed at the array's power supply connection between V_{CC} and GND. The bulk capacitor will overcome voltage slumps caused by PC board trace inductances.

V_{PP} Trace on Printed Circuit Boards

Programming flash memories, while they reside in the target system, requires that the printed circuit board designer pay attention to the V_{PP} power supply trace. The V_{PP} pin supplies the memory cell current for programming. Use similar trace widths and layout considerations given to the V_{CC} power bus. Adequate V_{PP} supply traces and decoupling will decrease V_{PP} voltage spikes and overshoots.

V_{CC} , V_{PP} , $RP\#$ Transitions and the Command/Status Registers

Programming and erase completion are not guaranteed if V_{PP} drops below V_{PPH} . If the V_{PP} Status bit of the Status Register (SR.3) is set to "1", a Clear Status Register command MUST be issued before further program/erase attempts are allowed by the WSM. Otherwise, the Program (SR.4) or Erase (SR.5) Status bits of the Status Register will be set to "1" if error is detected. $RP\#$ transitions to V_{IL} during program and erase also abort the operations. Data is partially altered in either case, and the command sequence must be repeated after normal operation is restored. Device poweroff, or $RP\#$ transitions to V_{IL} , clear the Status Register to initial value 80H.

The Command Register latches commands as issued by system software and is not altered by V_{PP} or $CE\#$ transitions or WSM actions. Its state upon powerup, after exit from Deep-Powerdown or after V_{CC} transitions below V_{LKO} , is FFH, or Read Array Mode.

must be reset to read array mode via the Read Array command if access to the memory array is desired.

Power Up/Down Protection

The 28F001BX is designed to offer protection against accidental erasure or programming during power transitions. Upon power-up, the 28F001BX is indifferent as to which power supply, V_{PP} or V_{CC} , powers up first. Power supply sequencing is not required. Internal circuitry in the 28F001BX ensures that the Command Register is reset to Read Array mode on power up.

A system designer must guard against spurious writes for V_{CC} voltages above V_{LKO} when V_{PP} is active. Since both $WE\#$ and $CE\#$ must be low for a command write, driving either to V_{IH} will inhibit writes. The Command Register architecture provides an added level of protection since alteration of memory contents only occurs after successful completion of the two-step command sequences.

Finally, the device is disabled, until $RP\#$ is brought to V_{IH} , regardless of the state of its control inputs. This provides an additional level of protection.

28F001BX Power Dissipation

When designing portable systems, designers must consider battery power consumption not only during device operation, but also for data retention during system idle time. Flash nonvolatility increases usable battery life because the 28F001BX does not consume any power to retain code or data when the system is off.

In addition, the 28F001BX's Deep-Powerdown mode ensures extremely low power dissipation even when system power is applied. For example, laptop and other PC applications, after copying BIOS to DRAM, can lower $RP\#$ to V_{IL} , producing negligible power consumption. If access to the boot code is again needed, as in case of a system RESET#, the part can again be accessed, following the t_{PHAV} wakeup cycle required after $RP\#$ is first raised back to V_{IH} . The first address presented to the device while in powerdown requires time t_{PHAV} , after $RP\#$ transitions high, before outputs are valid. Further accesses follow normal timing. See AC Characteristics—Read-Only Operations and Figure 12 for more information.

ABSOLUTE MAXIMUM RATINGS

Operating Temperature

- During Read 0°C to 70°C(1)
- During Erase/Program 0°C to 70°C(1)

Operating Temperature

- During Read -40°C to +85°C(2)
- During Erase/Program -40°C to +85°C(2)

Temperature under Bias -10°C to 80°C(1)

Temperature under Bias -20°C to +90°C(2)

Storage Temperature -65°C to 125°C

Voltage on Any Pin

- (except A₉, RP#, OE#, V_{CC} and V_{PP})
- with Respect to GND -2.0V to 7.0V(3)

Voltage on A₉, RP#, and OE#

- with Respect to GND -2.0V to 13.5V(3, 4)

V_{PP} Program Voltage

- with Respect to GND
- During Erase/Program -2.0V to 14.0V(3, 4)

V_{CC} Supply Voltage

- with Respect to GND -2.0V to 7.0V(3)

Output Short Circuit Current 100 mA(5)

NOTES:

1. Operating temperature is for commercial product defined by this specification.
2. Operating temperature is for extended temperature product defined by this specification.
3. Minimum DC voltage is -0.5V on input/output pins. During transitions, this level may undershoot to -2.0V for periods <20 ns. Maximum DC voltage on input/output pins is V_{CC} + 0.5V which, during transitions, may overshoot to V_{CC} + 2.0V for periods <20 ns.
4. Maximum DC voltage on A₉ or V_{PP} may overshoot to +14.0V for periods <20 ns.
5. Output shorted for no more than one second. No more than one output shorted at a time.

OPERATING CONDITIONS

Symbol	Parameter	Min	Max	Unit
T _A	Operating Temperature(1)	0	70	°C
T _A	Operating Temperature(2)	-40	85	°C
V _{CC}	Supply Voltage	4.50	5.50	V

DC CHARACTERISTICS V_{CC} = 5.0V ± 10%, T_A = 0°C to +70°C

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Conditions
I _{IL}	Input Load Current	1			± 1.0	μA	V _{CC} = V _{CC} Max V _{IN} = V _{CC} or GND
I _{LO}	Output Leakage Current	1			± 10	μA	V _{CC} = V _{CC} Max V _{OUT} = V _{CC} or GND
I _{CCS}	V _{CC} Standby Current			1.2	2.0	mA	V _{CC} = V _{CC} Max CE# = RP# = V _{IH}
				30	100	μA	V _{CC} = V _{CC} Max CE# = RP# = V _{CC} ± 0.2V
I _{CCD}	V _{CC} Deep-Powerdown Current	1		0.05	1.0	μA	RP# = GND ± 0.2V

NOTES: This is a production data sheet. The specifications are subject to change without notice.

*WARNING: Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.

DC CHARACTERISTICS $V_{CC} = 5.0V \pm 10\%$, $T_A = 0^\circ C$ to $+70^\circ C$ (Continued)

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Conditions
I_{CCR}	V_{CC} Read Current	1		13	30	mA	$V_{CC} = V_{CC} \text{ Max}$, $CE\# = V_{IL}$ $f = 8 \text{ MHz}$, $I_{OUT} = 0 \text{ mA}$
I_{CCP}	V_{CC} Programming Current	1		5	20	mA	Programming in Progress
I_{CCE}	V_{CC} Erase Current	1		6	20	mA	Erase in Progress
I_{CCES}	V_{CC} Erase Suspend Current	1, 2		5	10	mA	Erase Suspended $CE\# = V_{IH}$
I_{PPS}	V_{PP} Standby Current	1		± 1	± 10	μA	$V_{PP} \leq V_{CC}$
				90	200	μA	$V_{PP} > V_{CC}$
I_{PPD}	V_{PP} Deep-Powerdown Current	1		0.80	1.0	μA	$RP\# = GND \pm 0.2V$
I_{PPP}	V_{PP} Programming Current	1		6	30	mA	$V_{PP} = V_{PPH}$ Programming in Progress
I_{PPE}	V_{PP} Erase Current	1		6	30	mA	$V_{PP} = V_{PPH}$ Erase in Progress
I_{PPES}	V_{PP} Erase Suspend Current	1		90	300	μA	$V_{PP} = V_{PPH}$ Erase Suspended
I_{ID}	A_9 Intelligent Identifier Current	1		90	500	μA	$A_9 = V_{ID}$
V_{IL}	Input Low Voltage		-0.5		0.8	V	
V_{IH}	Input High Voltage		2.0		$V_{CC} + 0.5$	V	
V_{OL}	Output Low Voltage				0.45	V	$V_{CC} = V_{CC} \text{ Min}$ $I_{OL} = 5.8 \text{ mA}$
V_{OH}	Output High Voltage		2.4			V	$V_{CC} = V_{CC} \text{ Min}$ $I_{OH} = 2.5 \text{ mA}$
V_{ID}	A_9 Intelligent Identifier Voltage		11.5		13.0	V	
V_{PPL}	V_{PP} during Normal Operations	3	0.0		6.5	V	
V_{PPH}	V_{PP} during Prog/Erase Operations		11.4	12.0	12.6	V	
V_{LKO}	V_{CC} Erase/Write Lock Voltage		2.5			V	
V_{HH}	$RP\#$, $OE\#$ Unlock Voltage		11.4		12.6	V	Boot Block Prog/Erase

DC CHARACTERISTICS $V_{CC} = 5.0V \pm 10\%$, $T_A = -40^\circ C$ to $+85^\circ C$

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Conditions
I_{IL}	Input Load Current	1			± 1.0	μA	$V_{CC} = V_{CC} \text{ Max}$ $V_{IN} = V_{CC}$ or GND
I_{LO}	Output Leakage Current	1			± 10	μA	$V_{CC} = V_{CC} \text{ Max}$ $V_{OUT} = V_{CC}$ or GND
I_{CCS}	V_{CC} Standby Current			1.2	2.0	mA	$V_{CC} = V_{CC} \text{ Max}$ $CE\# = RP\# = V_{IH}$
				30	150	μA	$V_{CC} = V_{CC} \text{ Max}$ $CE\# = RP\# = V_{CC} \pm 0.2V$

DC CHARACTERISTICS $V_{CC} = 5.0V \pm 10\%$, $T_A = -40^\circ C$ to $+85^\circ C$ (Continued)

Symbol	Parameter	Notes	Min	Typ	Max	Unit	Test Conditions
I_{CCD}	V_{CC} Deep-Powerdown Current	1		0.05	2.0	μA	$RP\# = GND \pm 0.2V$
I_{CCR}	V_{CC} Read Current	1		13	35	mA	$V_{CC} = V_{CC} \text{ Max}$, $CE\# = V_{IL}$ $f = 8 \text{ MHz}$, $I_{OUT} = 0 \text{ mA}$
I_{CCP}	V_{CC} Programming Current	1		5	20	mA	Programming in Progress
I_{CCE}	V_{CC} Erase Current	1		6	20	mA	Erase in Progress
I_{CCES}	V_{CC} Erase Suspend Current	1, 2		5	10	mA	Erase Suspended $CE\# = V_{IH}$
I_{PPS}	V_{PP} Standby Current	1		± 1	± 10	μA	$V_{PP} \leq V_{CC}$
				90	400	μA	$V_{PP} > V_{CC}$
I_{PPD}	V_{PP} Deep-Powerdown Current	1		0.80	1.0	μA	$RP\# = GND \pm 0.2V$
I_{PPP}	V_{PP} Programming Current	1		6	30	μA	$V_{PP} = V_{PPH}$ Programming in Progress
I_{PPE}	V_{PP} Erase Current	1		6	30	mA	$V_{PP} = V_{PPH}$ Erase in Progress
I_{PPES}	V_{PP} Erase Suspend Current	1		90	400	μA	$V_{PP} = V_{PPH}$ Erase Suspended
I_{ID}	A_9 Intelligent Identifier Current	1		90	500	μA	$A_9 = V_{ID}$
V_{IL}	Input Low Voltage		-0.5		0.8	V	
V_{IH}	Input High Voltage		2.0		$V_{CC} + 0.5$	V	
V_{OL}	Output Low Voltage				0.45	V	$V_{CC} = V_{CC} \text{ Min}$ $I_{OL} = 5.8 \text{ mA}$
V_{OH}	Output High Voltage		2.4			V	$V_{CC} = V_{CC} \text{ Min}$ $I_{OH} = 2.5 \text{ mA}$
V_{ID}	A_9 Intelligent Identifier Voltage		11.5		13.0	V	
V_{PPL}	V_{PP} during Normal Operations	3	0.0		6.5	V	
V_{PPH}	V_{PP} during Prog/Erase Operations		11.4	12.0	12.6	V	
V_{LKO}	V_{CC} Erase/Write Lock Voltage		2.5			V	
V_{HH}	$RP\#$, $OE\#$ Unlock Voltage		11.4		12.6	V	Boot Block Prog/Erase

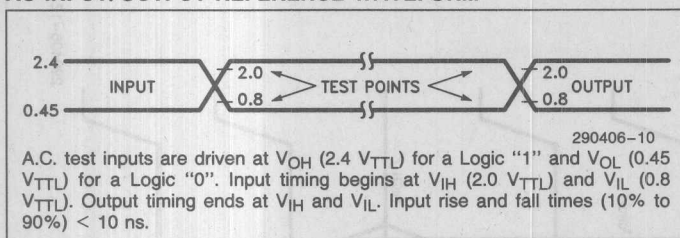
CAPACITANCE⁽⁴⁾ $T_A = 25^\circ C$, $f = 1 \text{ MHz}$

Symbol	Parameter	Max	Unit	Conditions
C_{IN}	Input Capacitance	8	pF	$V_{IN} = 0V$
C_{OUT}	Output Capacitance	12	pF	$V_{OUT} = 0V$

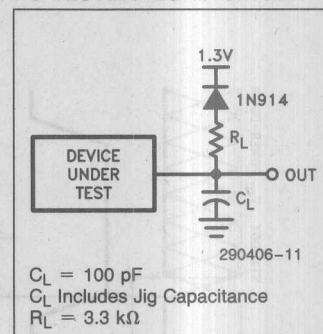
NOTES:

1. All currents are in RMS unless otherwise noted. Typical values at $V_{CC} = 5.0V$, $V_{PP} = 12.0V$, $T_A = 25^\circ C$. These currents are valid for all product versions (packages and speeds).
2. I_{CCES} is specified with the device deselected. If the 28F001BX is read while in Erase Suspend mode, current draw is the sum of I_{CCES} and I_{CCR} .
3. Erase/Programs are inhibited when $V_{PP} = V_{PPL}$ and not guaranteed in the range between V_{PPH} and V_{PPL} .
4. Sampled, not 100% tested.

AC INPUT/OUTPUT REFERENCE WAVEFORM



AC TESTING LOAD CIRCUIT



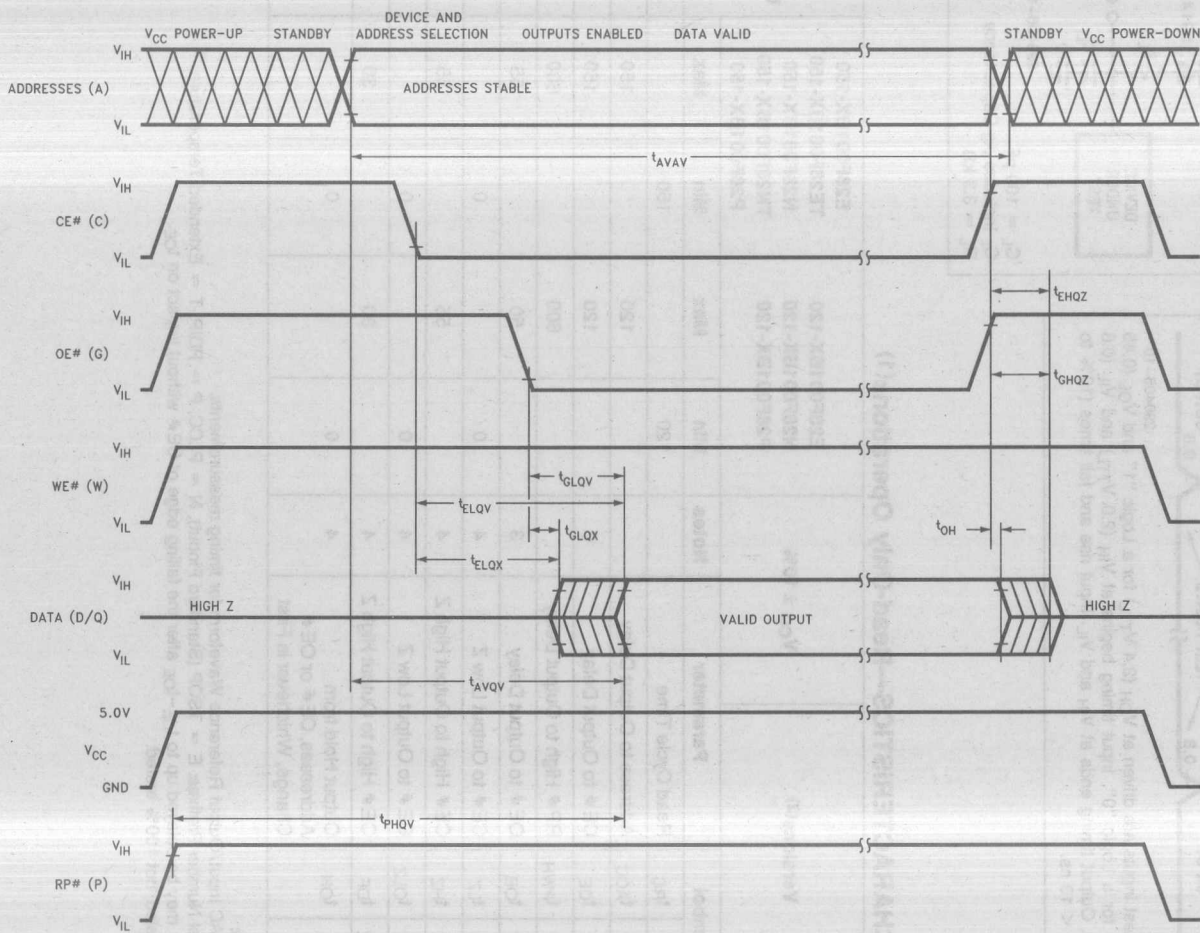
AC CHARACTERISTICS—Read-Only Operations(1)

Versions(2)		$V_{CC} \pm 10\%$		E28F001BX-120 N28F001BX-120 P28F001BX-120		E28F001BX-150 TE28F001BX-150 N28F001BX-150 TN28F001BX-150 P28F001BX-150		Unit
Symbol		Parameter	Notes	Min	Max	Min	Max	
tAVAV	tRC	Read Cycle Time		120		150		ns
tAVQV	tACC	Address to Output Delay			120		150	ns
tELQV	tCE	CE # to Output Delay	3		120		150	ns
tPHQV	tPWH	RP # High to Output Delay			600		600	ns
tGLQV	tOE	OE # to Output Delay	3		50		55	ns
tELQX	tLZ	CE # to Output Low Z	4	0		0		ns
tEHQZ	tHZ	CE # High to Output High Z	4		55		55	ns
tGLQX	tOLZ	OE # to Output Low Z	4	0		0		ns
tGHQZ	tDF	OE # High to Output High Z	4		30		30	ns
	tOH	Output Hold from Addresses, CE # or OE # Change, Whichever is First	4	0		0		ns

NOTES:

1. See AC Input/Output Reference Waveform for timing measurements.
2. Model Number Prefixes: E = TSOP (Standard Pinout), N = PLCC, P = PDIP, T = Extended Temperature.
3. OE # may be delayed up to $t_{CE} - t_{OE}$ after the falling edge of CE # without impact on t_{CE} .
4. Sampled, not 100% tested.

Figure 12. AC Waveform for Read Operations



AC CHARACTERISTICS—Write/Erase/Program Operations(1, 9)

Versions			V _{CC} ± 10%		28F001BX-120		28F001BX-150		Unit
Symbol		Parameter	Notes	Min	Max	Min	Max		
t _{AVAV}	t _{WC}	Write Cycle Time		120		150		ns	
t _{PHWL}	t _{PS}	RP # High Recovery to WE # Going Low	2	480		480		ns	
t _{ELWL}	t _{CS}	CE # Setup to WE # Going Low		10		10		ns	
t _{WLWH}	t _{WP}	WE # Pulse Width		50		50		ns	
t _{PHHWH}	t _{PHS}	RP # V _{HH} Setup to WE # Going High	2	100		100		ns	
t _{VPWH}	t _{VPS}	V _{PP} Setup to WE # Going High	2	100		100		ns	
t _{AVWH}	t _{AS}	Address Setup to WE # Going High	3	50		50		ns	
t _{DVWH}	t _{DS}	Data Setup to WE # Going High	4	50		50		ns	
t _{WHDX}	t _{DH}	Data Hold from WE # High		10		10		ns	
t _{WHAX}	t _{AH}	Address Hold from WE # High		10		10		ns	
t _{WHEH}	t _{CH}	CE # Hold from WE # High		10		10		ns	
t _{WHWL}	t _{WPH}	WE # Pulse Width High		50		50		ns	
t _{WHQV1}		Duration of Programming Operation	5, 6, 7	15		15		μs	
t _{WHQV2}		Duration of Erase Operation (Boot)	5, 6, 7	1.3		1.3		sec	
t _{WHQV3}		Duration of Erase Operation (Parameter)	5, 6, 7	1.3		1.3		sec	
t _{WHQV4}		Duration of Erase Operation (Main)	5, 6, 7	3.0		3.0		sec	
t _{WHGL}		Write Recovery before Read		0		0		μs	
t _{QVVL}	t _{VPH}	V _{PP} Hold from Valid SRD	2, 6	0		0		ns	
t _{QVPH}	t _{PHH}	RP # V _{HH} Hold from Valid SRD	2, 7	0		0		ns	
t _{PHBR}		Boot-Block Relock Delay	2		100		100	ns	

PROM Programmer Specifications

Versions		$V_{CC} \pm 10\%$		28F001BX-120		28F001BX-150		Unit
Symbol		Parameter		Notes	Min	Max	Min	
t_{GHHWL}		OE# V_{HH} Setup to WE# Going Low		2, 8	480		480	ns
t_{WHGH}		OE# V_{HH} Hold from WE# High		2, 8	480		480	ns

NOTES:

- Read timing characteristics during erase and program operations are the same as during read-only operations. Refer to AC Characteristics for Read-Only Operations.
- Sampled, not 100% tested.
- Refer to Table 3 for valid A_{IN} for byte programming or block erasure.
- Refer to Table 3 for valid D_{IN} for byte programming or block erasure.
- The on-chip Write State Machine incorporates all program and erase system functions and overhead of standard Intel Flash Memory, including byte program and verify (programming) and block precondition, precondition verify, erase and erase verify (erasing).
- Program and erase durations are measured to completion ($SR.7 = 1$). V_{PP} should be held at V_{PPH} until determination of program/erase success ($SR.3/4/5 = 0$).
- For boot block programming and erasure, RP# should be held at V_{HH} until determination of program/erase success ($SR.3/4/5 = 0$).
- Alternate boot block access method.
- Erase/Program Cycles on extended temperature products is 10,000 cycles.

ERASE AND PROGRAMMING PERFORMANCE

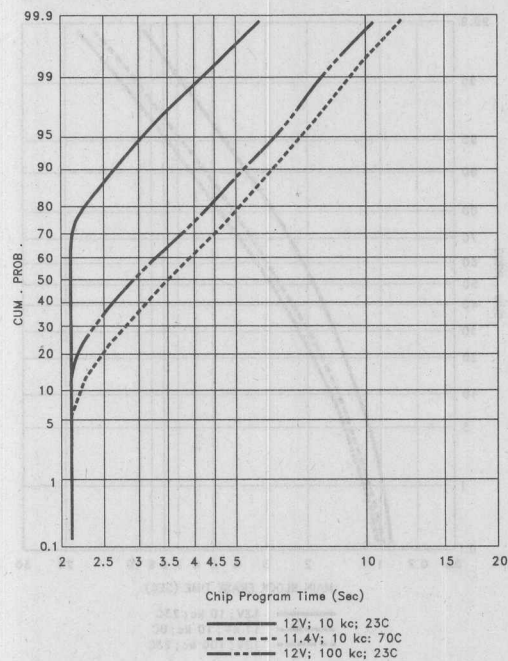
Parameter	Notes	28F001BX-120			28F001BX-150			Unit
		Min	Typ ⁽¹⁾	Max	Min	Typ ⁽¹⁾	Max	
Boot Block Erase Time	2		2.10	14.9		2.10	14.9	Sec
Boot Block Program Time	2		0.15	0.52		0.15	0.52	Sec
Parameter Block Erase Time	2		2.10	14.6		2.10	14.6	Sec
Parameter Block Program Time	2		0.07	0.26		0.07	0.26	Sec
Main Block Erase Time	2		3.80	20.9		3.80	20.9	Sec
Main Block Program Time	2		2.10	7.34		2.10	7.34	Sec
Chip Erase Time	2		10.10	65		10.10	65	Sec
Chip Program Time	2		2.39	8.38		2.39	8.38	Sec

NOTES:

1. 25°C, 12.0 V_{pp}.
2. Excludes System-Level Overhead.

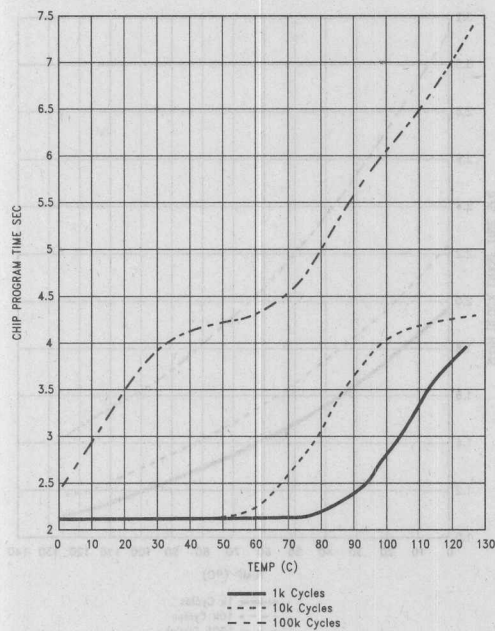
Symbol	Parameter	Notes	V _{CC} ± 10%			Unit
			Min	Max	Typ	
t _{SHW}	CE ₀ V _{PH} Setup to WE ₀ Going Low	2.8	480		480	ns
t _{WHC}	CE ₀ V _{PH} Hold from WE ₀ High	2.8	480		480	ns

- NOTES:
1. Read timing characteristics during erase and program operations are the same as during read-only operations. Refer to the 28F001BX-T/28F001BX-B data sheet for Read-Only Operations.
 2. Erase/program time is measured to completion (SRV = 1). V_{CC} should be held at V_{PH} until termination of program/erase and erase. R_{PH} should be held at V_{PH} until termination of program/erase and erase. (SRV = 0).
 3. Refer to Table 1 for valid A₀ for byte programming or block erase.
 4. Refer to Table 1 for valid Q₀ for byte programming or block erase.
 5. The on-chip write data machine insures all program and erase system functions and overwrites of standby data.
 6. Flash memory reading byte program and verify (programming) and block precondition, precondition, verify, erase and erase verify (erasing).
 7. For block programming and erase, R_{PH} should be held at V_{PH} until termination of program/erase and erase. (SRV = 0).
 8. Alternate boot block access method.
 9. Erase/program/erase on extended temperature products is 10,000 cycles.



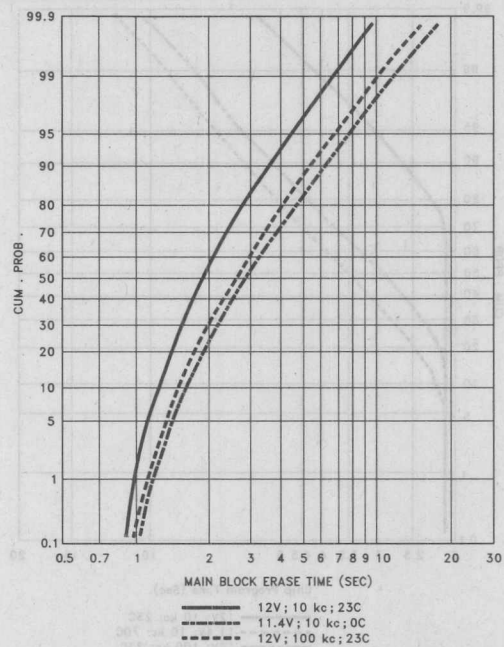
290406-19

Figure 13. 28F001BX Typical Programming Capability



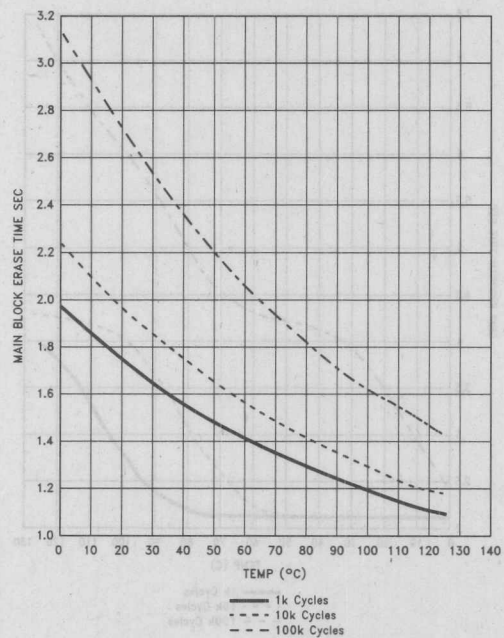
290406-20

Figure 14. 28F001BX Typical Programming Time at 12V



290406-21

Figure 15. 28F001BX Typical Erase Capability



290406-22

Figure 16. 28F001BX Typical Erase Time at 12V

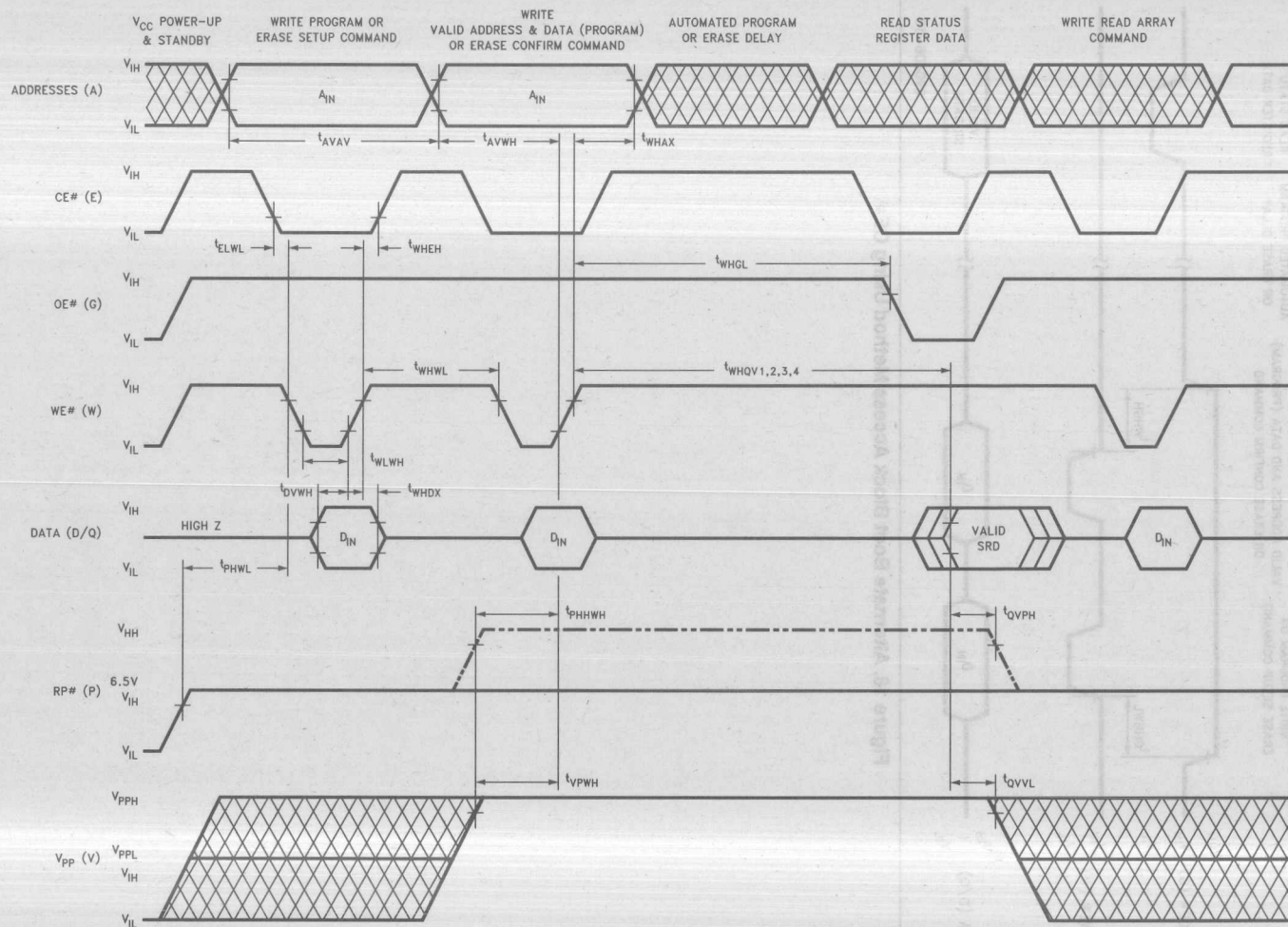


Figure 17. AC Waveform for Write Operations

ALTERNATE CE #-CONTROLLED WRITES(1)

Versions			V _{CC} ± 10%		28F001BX-120		28F001BX-150		Unit
Symbol		Parameter	Notes	Min	Max	Min	Max		
t _{AVAV}	t _{WC}	Write Cycle Time		120		150		ns	
t _{PHEL}	t _{PS}	RP# High Recovery to CE# Going Low	2	480		480		ns	
t _{WLEL}	t _{WS}	WE# Setup to CE# Going low		0		0		ns	
t _{ELEH}	t _{CP}	CE# Pulse Width		70		70		ns	
t _{PHHEH}	t _{PHS}	RP# V _{HH} Setup to CE# Going High	2	100		100		ns	
t _{VPEH}	t _{VPS}	V _{PP} Setup to CE# Going High	2	100		100		ns	
t _{AVEH}	t _{AS}	Address Setup to CE# Going High	3	50		50		ns	
t _{DVEH}	t _{DS}	Data Setup to CE# Going High	4	50		50		ns	
t _{EHDX}	t _{DH}	Data Hold from CE# High		10		10		ns	
t _{EHAX}	t _{AH}	Address Hold from CE# High		15		15		ns	
t _{EHHW}	t _{WH}	WE# Hold from CE# High		0		0		ns	
t _{EHEL}	t _{EPH}	CE# Pulse Width High		25		25		ns	
t _{EHQV1}		Duration of Programming Operation	5, 6	15		15		μs	
t _{EHQV2}		Duration of Erase Operation (Boot)	5, 6	1.3		1.3		sec	
t _{EHQV3}		Duration of Erase Operation (Parameter)	5, 6	1.3		1.3		sec	
t _{EHQV4}		Duration of Erase Operation (Main)	5, 6	3.0		3.0		sec	
t _{EHGL}		Write Recovery before Read		0		0		μs	
t _{QVVL}	t _{VPH}	V _{PP} Hold from Valid SRD	2, 5	0		0		ns	
t _{QVPH}	t _{PHH}	RP# V _{HH} Hold from Valid SRD	2, 6	0		0		ns	
t _{PHBR}		Boot-Block Relock Delay	2		100		10	ns	

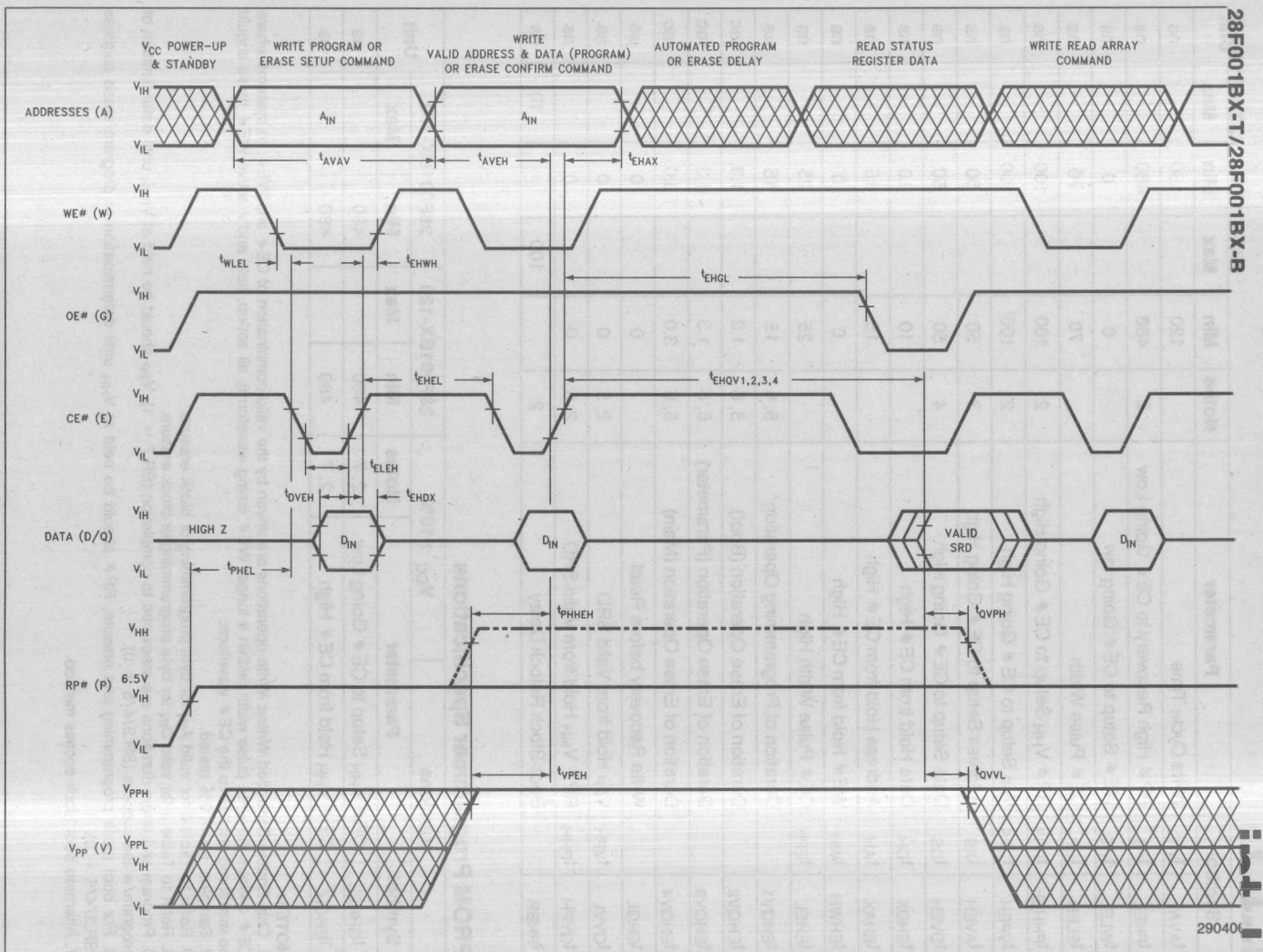
PROM Programmer Specifications

Versions		V _{CC} ± 10%	28F001BX-120		28F001BX-150		Unit
Symbol	Parameter	Notes	Min	Max	Min	Max	
t _{GHHEL}	OE # V _{HH} Setup to CE # Going Low	2, 7	480		480		ns
t _{EHGH}	OE # V _{HH} Hold from CE # High	2, 7	480		480		ns

NOTES:

1. Chip-Enable Controlled Writes: Write operations are driven by the valid combination of CE# and WE#. In systems where CE# defines the write pulse width (within a longer WE# timing waveform), all set-up, hold and inactive WE# times should be measured relative to the CE# waveform.
2. Sampled, not 100% tested.
3. Refer to Table 3 for valid A_{IN} for byte programming or block erasure.
4. Refer to Table 3 for valid D_{IN} for byte programming or block erasure.
5. Program and erase durations are measured to completion ($SR.7 = 1$). V_{PP} should be held at V_{PPH} until determination of program/erase success ($SR.3/4/5 = 0$).
6. For boot block programming and erasure, RP# should be held at V_{HH} until determination of program/erase success ($SR.3/4/5 = 0$).
7. Alternate boot block access method.

Figure 19. Alternate AC Waveform for Write Operations



ORDERING INFORMATION

T	P	2	8	F	0	0	1	B	X	-	T	1	2	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

PACKAGE

E STANDARD 32 LEAD TSOP

N 32 LEAD PLCC

P 32-PIN PLASTIC DIP

TEMPERATURE RANGE

T = EXTENDED (-40°C to +85°C)

BLANK = COMMERCIAL (0°C to +70°C)

ACCESS SPEED (ns)

120 ns

150 ns

T TOP BOOT DEVICE

B BOTTOM BOOT DEVICE

290406-18

VALID COMBINATIONS:

E28F001BX-T120 N28F001BX-T120 P28F001BX-T120

E28F001BX-T150 N28F001BX-T150 P28F001BX-T150

TE28F001BX-T150 TN28F001BX-T150

E28F001BX-B120 N28F001BX-B120 P28F001BX-B120

E28F001BX-B150 N28F001BX-B150 P28F001BX-B150

TE28F001BX-B150 TN28F001BX-B150

ADDITIONAL INFORMATION

	Order Number		Order Number
ER-20 "ETOX II Flash Memory Technology"	294005	AP-316 "Using Flash Memory for In-System Reprogrammable Nonvolatile Storage"	292046
RR-60 "ETOX II Flash Memory Reliability Data Summary"	293002	AP-341 "Designing an Updatable BIOS Using Flash Memory"	292077

4

REVISION HISTORY

Number	Description
-004	Removed Preliminary classification. Latched address A ₁₆ in Figure 5. Updated Boot Block Program and Erase section: "If boot block program or erase is attempted while RP # is at V _{IH} , either the Program Status or Erase Status bit will be set to "1", reflective of the operation being attempted and indicating boot block lock. " Updated Figure 11, 28F001BX Erase Suspend/Resume Flowchart Added DC Characteristics typical current values Combined V _{PP} Standby current and V _{PP} Read current into one V _{PP} Standby current spec with two test conditions (DC Characteristics table) Added maximum program/erase times to Erase and Programming Performance table. Added Figures 13-16 Added Extended Temperature proliferations
-005	PWD changed to RP # for JEDEC standardization compatibility Revised symbols, i.e.; \overline{CE} , \overline{OE} , etc. to CE #, OE #, etc.

APPLICATION NOTE

Designing an Updatable BIOS Using FLASH Memory

BRIAN DIPERT
DON VERNER
MCD MARKETING APPLICATIONS

October 1993

Designing an Updatable BIOS Using Flash Memory

CONTENTS PAGE

1.0 INTRODUCTION 4-210

2.0 FLASH MEMORY 4-210

2.1 EPROM Roots; Review of Flash Process vs. EPROM & EEPROM .. 4-210

2.2 Program and Erase Automation .. 4-210

2.3 Blocked Architecture 4-211

2.4 Deep Powerdown Mode 4-212

2.5 28F001BX Pinouts, Physical Layout and Upgrade 4-213

Plastic Dual-In Line

Plastic Leaded Chip Carrier

Thin Small Outline Package

2.6 Vpp Specifications 4-216

Fixed Vpp and VCC

RP#, VCC and Vpp Lockout Protection

3.0 HARDWARE DESIGN CONSIDERATIONS 4-217

3.1 BIOS Boot Code Requirements and System Configurations 4-218

Address Shift Configuration

Address Inversion Configuration

3.2 Vpp Generation 4-220

Using System 12V Directly

Pumping 5V to 12V

Security

Using a MOSFET Switch

3.3 Modifying an Existing Motherboard 4-221

EPROM/ROM Designs

28F010 Flash Memory Designs

3.4 In-System Write vs. On-Board Programming 4-222

3.5 Ideas for Using Extra Adaptor Space 4-223

CONTENTS PAGE

4.0 SOFTWARE DESIGN CONSIDERATIONS 4-223

4.1 Update Software for a Modified System 4-224

4.2 Pseudo-Code Overview 4-225

Pseudo-Code for Flash Update Routine

4.3 Initializing the System 4-225

Checking Power

4.4 Code Loader Routine 4-225

4.5 Flash Reprogramming Routines .. 4-226

On-Chip Erase Algorithm

Erase Suspend/Resume

On-Chip Programming Algorithm

Full Status Checks

4.6 Recovery Routine Overview 4-230

4.7 Power Management 4-230

5.0 SUMMARY 4-231

5.1 Traditional BIOS Storage and Disadvantages 4-231

5.2 Advantages of an Updatable BIOS 4-231

5.3 Advantages of Adding DOS in FLASH 4-231

5.4 Advantages of Adding 1 MB–4 MB of Resident Code Storage 4-231

APPENDIX A—Software Routines 4-232

APPENDIX B—MS-DOS ROM Version Overview 4-248

APPENDIX C—BIOS Vendor Information 4-249

APPENDIX D—Microprocessor/ Microcontroller Compatibility Chart 4-250

As PC computing platforms increase in complexity, so does the associated BIOS code. Sophisticated hardware and BIOS software increase the potential for revisions. Time-to-market goals require faster completion of designs from conception to production, leaving less time for new-peripheral BIOS support. As an example, many 80286-based PC/ATs lack BIOS support for 3 $\frac{1}{2}$ " floppy drives! Once a computer is out the door, code revisions are far more difficult and costly. Code revisions with EPROM require either a service call or sending EPROMs to the end user, assuming nothing else goes wrong in the process. The alternative to BIOS update is a prematurely obsolete system, unable to support new industry standards and peripheral systems.

Flash memory offers the same nonvolatile storage as EPROM, but additionally offers in-system write capability. Using Intel's 28F001BX for BIOS storage, code updates are done quickly in the factory during test and debug, while allowing cost-effective field updates to end users via floppy disks or modem BBS.

This application note describes various methods of implementing a flash memory BIOS using the 28F001BX. Design targets are both laptop and desktop systems. The primary emphasis is on application of flash memory for BIOS and ROM executable software applications. Detailed 28F001BX information is covered in the datasheet, available through your local sales office.

2.0 FLASH MEMORY

This section provides a brief overview of Intel's Flash Memory and in particular, Intel's 28F001BX blocked flash memory family. It covers the following:

- Flash memory's EPROM roots
- Program and Erase Automation
- Blocked Architecture
- Deep Powerdown Mode
- Pinouts, physical layout and upgrade for different packages
- V_{pp} specifications

Major features of the 28F001BX are in-system write, selective block erase, program/erase automation, SRAM-like command interface, deep powerdown capability, fixed V_{CC} and V_{pp} supplies and hardware lock-out protection.

2.1 EPROM Roots; Review of Flash Process vs EPROM & EEPROM

Intel's ETOX II (EPROM Tunnel OXide) flash memory is a single-transistor cell providing nonvolatile

trically erasing all data bits in parallel, then randomly programming data into any byte in the array. The programming operation is achieved via channel hot electron injection (CHE), just like EPROMs. Flash electrical erasure, however, is accomplished through Fowler-Nordheim (FN) tunneling. Using separate program and erase methods (CHE vs. FN Tunneling), in different cell locations, drain vs. source, permits process optimization for high cycling endurance—the number of complete erase and re-writes. Traditional low-density EEPROMs tunnel through the same memory cell junction for both programming and erasure. Because EEPROMs erase before programming each byte, these processes must occur very fast. Therefore, internal voltages used to program or erase 5V-only EEPROM memory cells are high (e.g., 18V–30V). The combination of higher voltage with programming and erasing through the same junction contributes to EEPROM's oxide breakdown, poor data retention and reduced cycling capability.

Intel's flash memory erasure (tunneling) voltage is below the critical oxide breakdown voltage. By using block erasure instead of EEPROM's byte erasure, erase times are relaxed, reducing tunneling voltages. Programming Intel's Flash Memory is non-destructive to the floating gate oxide compared to EEPROM's use of tunneling for programming. These features for erase and programming provide Intel's Flash Memory with the highest endurance (typically over 100K cycles) compared to that of traditional EEPROM cycling. Furthermore, flash memory exhibits lower failure rates at any given cycle count.

2.2 Program and Erase Automation

The 28F001BX integrates a Write State Machine (WSM) on-chip to internally implement program and erase algorithms. Operations are initiated through command sequence writes to the Command Register, and progress is reported back to the user through Status Register bits. Software timers are no longer required, as timing is now regulated through the on-chip oscillator. System software requirements are decreased in comparison to past algorithms, minimizing overhead and development effort, and allowing code execution and interrupt servicing while simultaneously programming or erasing the device.

The Erase Suspend command halts block erase to execute code or read data from any other block. This feature gives the system the capability to service higher level interrupts requiring data from the 28F001BX during the erase interval. After issuing the Erase Resume command, the WSM continues erase where it left off when suspended.

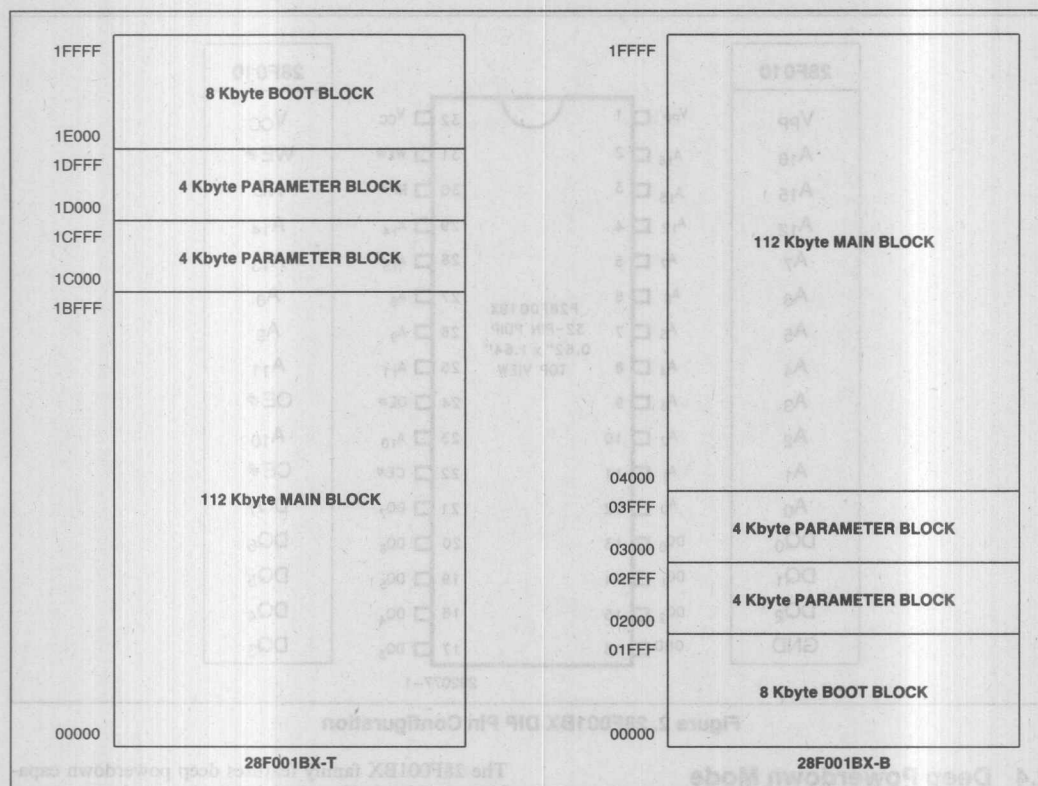


Figure 1. 28F001BX Memory Maps

2.3 Blocked Architecture

The 28F001BX family combines the safety of a hardware-protected 8 Kbyte "boot" block with the flexibility of two 4 Kbyte "parameter" blocks and one 112 Kbyte main block. Each block can be individually erased and programmed without affecting code stored in another block, ensuring data integrity. The boot block is intended to contain secure code which minimally will bring up the system and download code to the other blocks of the 28F001BX if required. Once programmed, it is hardware-locked from further alteration, guaranteeing true non-volatility.

The 28F001BX-T's lockable block location provides compatibility with microprocessors and microcontrollers that boot from the top of the memory map, such as many of Intel's microprocessor families. The seg-

mentation of the 28F001BX-B is identical. Its lockable block location provides compatibility with microprocessors that boot from low memory, such as Motorola and AMD products. See Figure 1 for illustrations of the two memory maps available in the 28F001BX family.

The two 4 Kbyte parameter blocks have multiple uses in BIOS environments. They can be used to back up the CMOS setup parameters such as floppy and hard disk type, processor speed, system memory size, graphic display type and presence of a coprocessor. Today, should the system battery fail, the user loses information in battery-backed SRAM. The non-volatile parameter blocks provide the system capability to automatically back up and recover this information. Also, EISA systems can store software variable information such as add-in board addresses, DMA channels and interrupt values/levels.

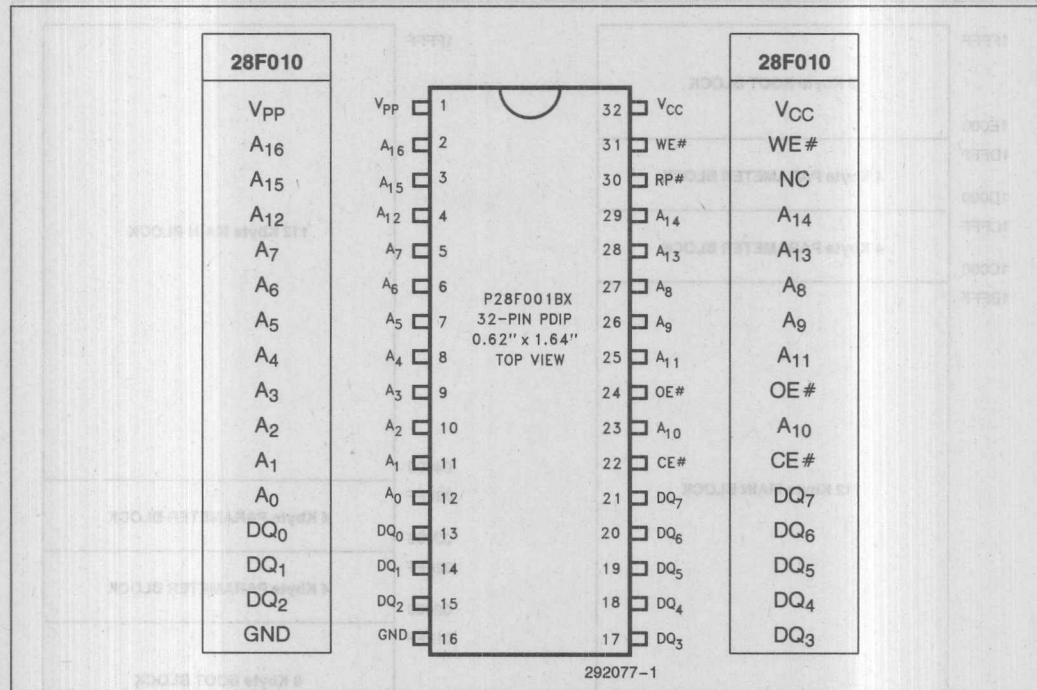


Figure 2. 28F001BX DIP Pin Configuration

2.4 Deep Powerdown Mode

Market analysts predict that the high-growth segments of the PC industry for the next several years will be in the laptop, palmtop and handheld product lines. Power management software is becoming an integral part of PC system BIOS as manufacturers attempt to squeeze the maximum amount of system operation time from their battery pack power supplies. A key indicator of this trend is Intel's i386TMSL microprocessor superset, which adds hardware power management to the Intel 386 architecture and answers the needs of low-power applications.

The 28F001BX family features deep powerdown capability and is ideally suited for these same battery-operated systems. Powerdown is entered through low voltage on the RP# pin. Typical power consumption through V_{CC} in powerdown is 0.25 μ W, regardless of power supply voltages and activity on the external bus. Once BIOS is shadowed to system DRAM for high-speed execution, the 28F001BX can be shut down for minimal current drain.

This same pin, with 12V present on it, gates program and erase of the boot block. Such a hardware lockout preserves the system boot code once initially programmed and protects it from inadvertent alteration or computer virus compromise.

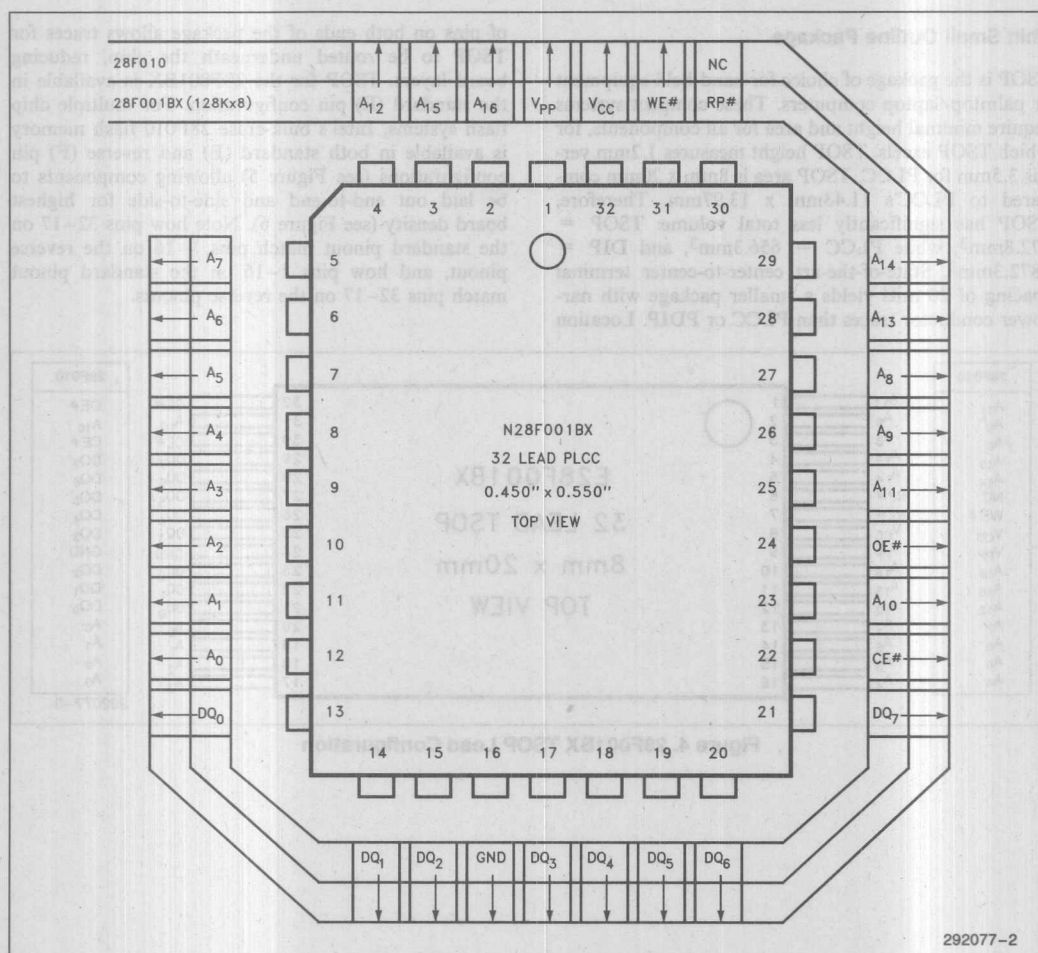


Figure 3. 28F001BX PLCC Lead Configuration

2.5 28F001BX Pinouts, Physical Layout and Upgrade

Intel's 28F001BX is offered in three standard 32-pin packages: Plastic Dual In-line Package (PDIP), Plastic Leaded Chip Carrier (PLCC), and Thin Small Outline Package (TSOP). All three pinouts provide backward compatibility with Intel's 28F010 bulk-erase flash. See Figures 2, 3, and 4 for pinout details.

Plastic Dual In-Line Package

PDIPs with sockets provide an excellent way to prototype and debug new designs. The 28F001BX is backward pin-compatible with 1 Mbit standard flash and EPROMs.

Plastic Leaded Chip Carrier

Most system designs today require surface mount technology (SMT) due to shrinking board real estate and portable form factors. PLCC is one SMT component that uses as little as 35% of the overall board space compared to PDIP. Its small size is attributed to the center-to-center lead spacing of 50 mils versus 100 mils, as well as its four-sided pinout. The J-lead design allows the PLCC to be directly soldered to the circuit board. Most SMT manufacturing equipment can easily handle the PLCC's 50-mil lead pitch. PLCC SMT sockets such as that offered by AMP (P/N 821977-1) have an identical foot-print for 32-pin devices. Such sockets can be used in place of directly soldering a PLCC for prototype build and code testing. Once the reprogramming code is tested and debugged, flash PLCCs can then be surface-mounted without socketing during production runs.

TSOP to be routed underneath the chip, reducing board layers. TSOP for the 28F001BX is available in the standard (E) pin configuration. For multiple chip flash systems, Intel's bulk-erase 28F010 flash memory is available in both standard (E) and reverse (F) pin configurations (see Figure 5) allowing components to be laid out end-to-end and side-to-side for highest board density (see Figure 6). Note how pins 32-17 on the standard pinout match pins 1-16 on the reverse pinout, and how pins 1-16 on the standard pinout match pins 32-17 on the reverse pinouts.



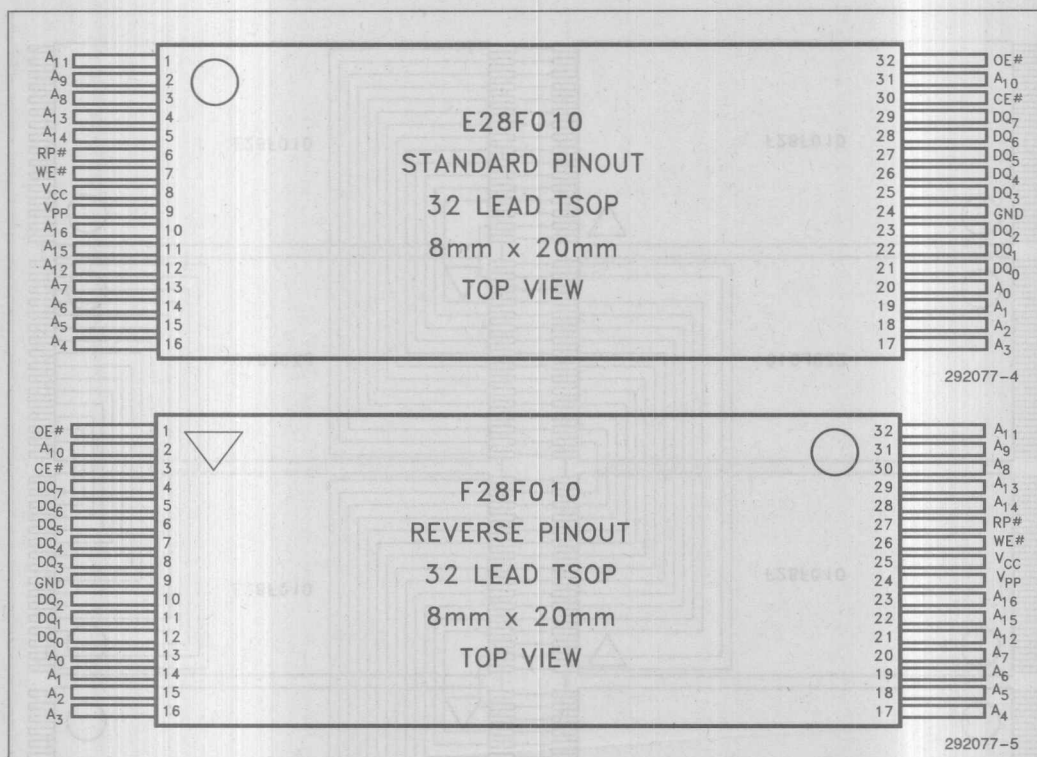


Figure 5. 28F010 Standard and Reverse TSOP Lead Configurations

4

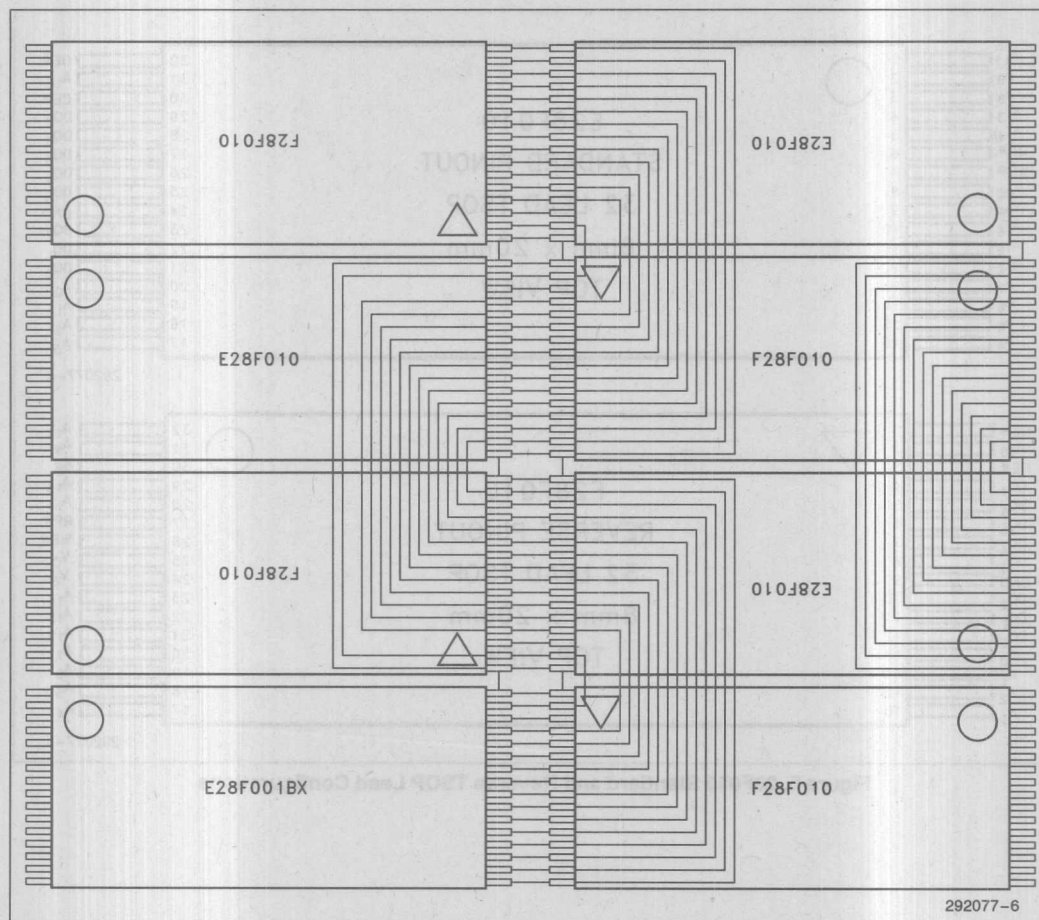


Figure 6. 28F001BX and 28F010's in Serpentine Layout Using TSOPs

2.6 V_{pp} Specifications

Fixed V_{pp} and V_{cc}

Flash memories, like EPROMs, require a 12V programming power supply. Unlike EPROMs, however, the V_{pp} for flash memory is a fixed, standard level. When combined with the Command Register erase/program control, Intel flash memories use a simple, SRAM-like hardware interface with standard microprocessor timings.

Intel's Flash Memory V_{pp} specification is 12.0V \pm 0.6V (5%), compatible with most off-the-shelf system power supplies. The IBM PC technical reference manual specifies the 12V power supply at 12.0V, +5% and -4%. Additionally, some hard and floppy drives require 12V \pm 5%. Therefore, most PC power supplies have 12V supplies with \pm 5% or better tolerance. Possi-

ble exceptions to this are laptop and/or palmtop PCs. Some of these require 5V-only designs, in which case 5V is charge-pumped to 12V. It is essential to use the specified V_{pp} when programming and erasing flash. Once the commands to program or erase are issued, the device internally derives the required voltage references from the V_{pp} supply. Therefore, an improper V_{pp} level degrades the performance of the part.

The Write State Machine automatically monitors the voltage present on the V_{pp} pin, beginning when program or erase setup commands are issued and continuing throughout the internal algorithm interval. If low V_{pp} is detected, the WSM automatically aborts the program or erase attempt and reports an improper voltage error to the user through the Status Register. The hardware design section discusses various methods of V_{pp} generator if your 12V power supply does not meet the proper tolerances or 12V is not available.

RP#, V_{CC} and V_{PP} Lockout Protection

The RP# (Reset/Powerdown) pin provides hardware write protection for 28F001BX flash memories. Until this pin transitions to TTL-level V_{IH}, write attempts to the device Command Register are ignored, regardless of power supply levels or activity on the system bus and control inputs. Typically, the system designer will gate this signal with a system POWER GOOD indicator output to ensure system stability before memory accesses begin.

The 28F001BX family provides additional protection for designs that tie 12V directly to the device. Since the 12V supply is less capacitively loaded than the 5V supply, the 12V power supply reaches full value faster during power-on. If Command Register lockout protection was not provided, a finite possibility exists that inadvertent writes may occur during power-on. For this case, Intel's 28F001BX flash memory supplies Command Register lockout protection when V_{CC} is below 2.5V, preventing writes to flash memory from occurring. Since CMOS logic is valid at 2.0V, a 0.5V margin of protection exists, providing extra time for control signals to settle before the Command Register is activated. Program/erase inhibit is guaranteed with V_{PP} below 6.5V, with corresponding V_{PP} low reported through the Status Register. Once V_{CC} reaches 2.5V, the Command Register begins processing valid commands, and program/erase attempts may initiate with V_{PP} greater than 6.5V. At this point, the system is responsible for write control.

When the 28F001BX V_{CC} powers up, or after the RP# pin transitions to V_{IL} and back to V_{IH}, the Command Register is automatically initialized to Read Array mode.

3.0 HARDWARE DESIGN CONSIDERATIONS

The system level hardware requirements for implementing BIOS and application storage in flash are:

- Write Enable available to all of the flash memory
- 12V routed to flash location or generated on-board
- CMOS control-signal interface, or RP# gated by a power-good signal
- Data buffer or transceiver that works in both write and read directions
- Space in memory map allocated for each application's size

Intel's i386SL microprocessor superset was chosen for the design example, shown in Figure 7. The Intel386SL microprocessor superset integrates all major components of a Intel386 based design on two chips, including bus memory, cache controllers and the ISA peripheral subsystem. Additionally, it consolidates hardware power management for battery-operated designs such as laptop, handheld and palmtop computers.

Note the clean interface between the superset and 28F001BX-T. Flash memory was comprehended early in the design of the Intel386SL microprocessor superset, to ensure a minimal-chip interface. Transceivers for the system bus, as well as a flash memory CE# signal, are integrated on the Intel386SL.

4

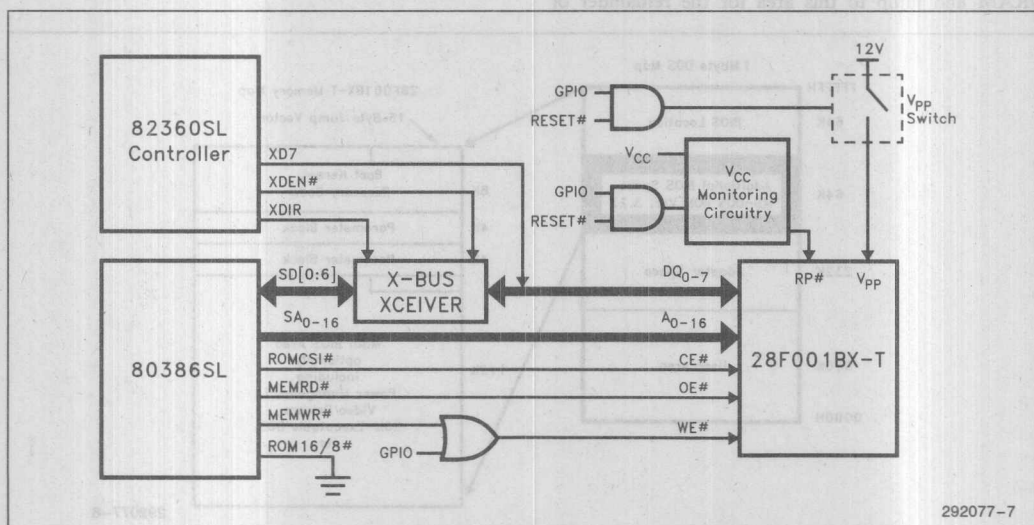


Figure 7. Intel386SL Microprocessor Superset with 28F001BX-T Flash BIOS

The RP# pin is gated by a power good signal to ensure control logic integrity before writes to the 28F001BX-T are allowed. It is also gated by System Reset, to abort program or erase if required for CPU reboot, and by a separate General Purpose Input-Output (GPIO) line to power down the device once BIOS is shadowed to RAM. CMOS logic will guarantee lowest power dissipation.

Similarly, system 12V is gated both by System Reset and a GPIO line. Software can switch 12V to the 28F001BX-T only when programming or erasing it, minimizing system power consumption. Vpp generation and switching methods are discussed in Section 3.3.

Application code, assuming a ROM in the BIOS socket, is sometimes designed to write to BIOS locations to generate software timing delays (versus using NOPs). Gating WR# to the flash memory with a GPIO line disables writes until desired by BIOS update software.

3.1 BIOS Boot Code Requirements and System Configurations

The previous design assumed that shadow RAM was available in the system. Referencing Figure 8, we see that the BIOS is actually stored in the main block of the 28F001BX, from system address E0000H–FBFFFFH. In this scenario, the processor jump vectors, BIOS checksum and recovery code are stored in the 8 Kbyte boot block. This is the area the processor will jump to on powerup or after reset. The boot block code will execute a checksum check of the main block for a valid BIOS. If successful, the processor will check system RAM, copy the main block code to high memory DRAM and jump to this area for the remainder of

Power On Self Test (POST), as well as further BIOS calls. Optionally, the 28F001BX can then be disabled for power savings.

If BIOS checksum determines an invalid BIOS, the system RAM and floppy drive (or possibly modem) are initialized using the boot block recovery code. The system requests (through screen display or speaker "beeps") that the user install the BIOS update floppy disk. A search of the floppy disk is made for a specific file name, and once found, update code is used to re-initialize the main BIOS block. System reboot restores normal operation. Alternatively, the BIOS recovery code can contain specific, non-DOS sector/track information pertaining to the location of the new BIOS update file. Thus, the file is protected and not readable to basic DOS users.

If ROM BIOS disable is overridden by system software or the user (through setup utility), the design must compensate for the altered BIOS location to prevent BIOS calls jumping to incorrect code locations. The following two methods provide alternative solutions for the system designer.

Address Shift Configuration

In this scenario, after BIOS initialization is complete, a write to a latch, register or flip-flop shifts addresses for future BIOS code fetches by 16 Kbytes. This allows the system to correctly access the main block and bypass parameter and boot blocks. A system reset or loss of power clears this latch, allowing booting from the boot block once again. Figure 9 shows the input and output signals required for a μ PLD address shifter. It shifts addresses in the range FFFFFH–E4000H (112 Kbytes) to FBFFFFH–E0000H.

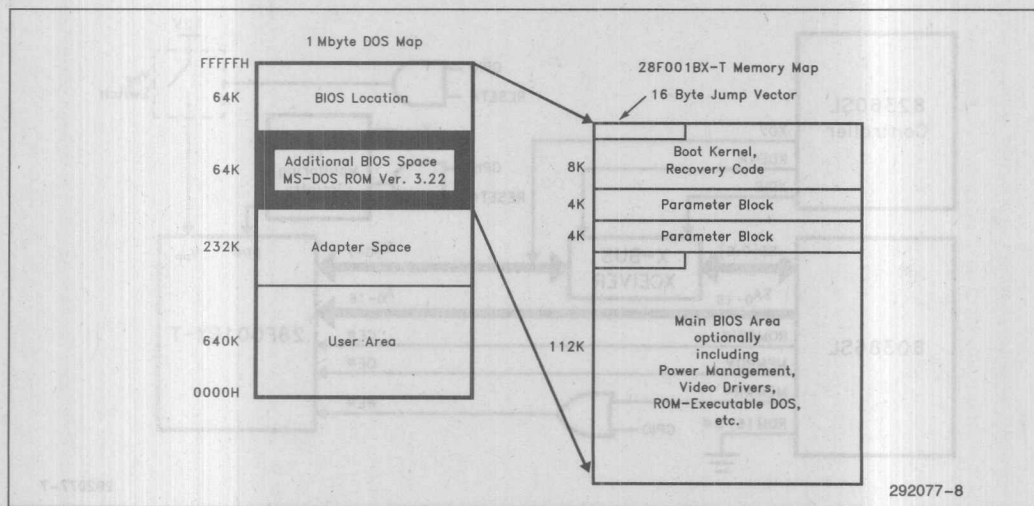


Figure 8. 28F001BX-T in 1 Mbyte DOS Memory Map

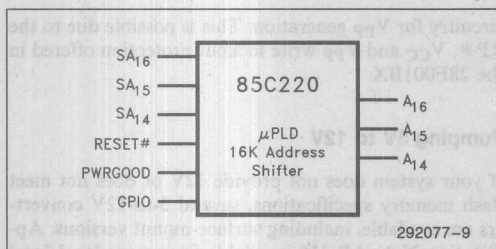


Figure 9. Address Shift Circuitry

Address Inversion Configuration

Figure 10 presents an alternative approach to configuring the 28F001BX in the system memory map. Simple inversion of address line A₁₆ to the 28F001BX moves the boot block to the lower half of flash memory as seen by the system. In normal operation, the processor boots and executes from the main array areas, which store the system BIOS, video BIOS and/or DOS in ROM.

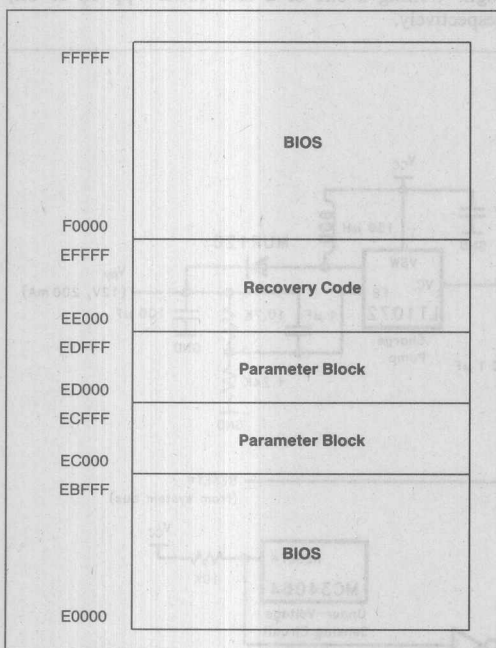


Figure 10. Inversion Configuration (Normal)

If power loss aborts a BIOS update, the main array block will be partially programmed/erased and the code in this block unusable. The system will “hang” or not boot at all. To boot from the boot recovery block, restore address A₁₆ polarity, producing the memory map shown in Figure 11. A keyboard sequence, switch on the back of the PC or jumper on the motherboard can toggle A₁₆ restore logic and “un-invert” it. After reconfiguration, the processor boots from the boot block and executes its recovery algorithm to restore main array block contents. Re-inverting A₁₆ reinstates normal system bootup and operation.

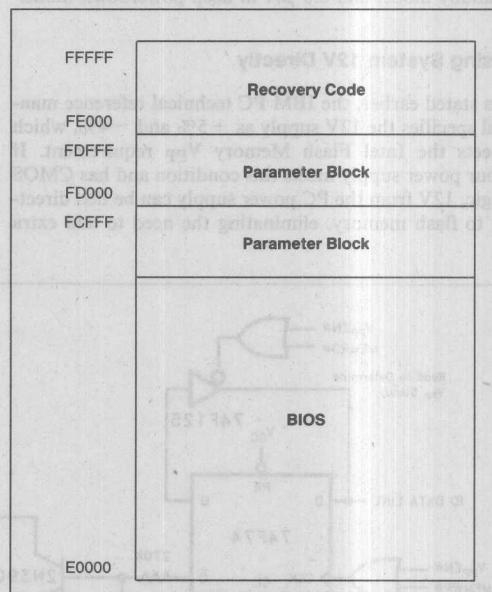


Figure 11. Inversion Configuration (Recovery)

Since standard BIOS code does not support boot block recovery, your BIOS software engineers must design the recovery code for the 8 Kbyte block. See Section 4.6 for a flowchart of an example recovery algorithm. Third-party BIOS vendors, working with Intel, have also developed recovery code for the 28F001BX (see Appendix C). With the exception of this recovery section, the rest of the BIOS remains the same.

Linear Technology's LT1072, a switching regulator, is used as a 5V to 12V charge pump. The 10.7K and 1.24K resistors are used to establish the correct reference voltage to obtain 12V. The 100 μ F capacitor at the output can handle up to 200 mA. For a single- or double-chip BIOS design, this capacitor value can be halved or even quartered to allow selection of a SMT capacitor value, since the maximum I_{pp} current per device is only 30 mA (10 mA typical). Allow sufficient time when switching V_{pp} on, letting the charge pump level out and enabling the Command Register to receive program or erase commands. The diode, MUR120, keeps the inductor from absorbing current from the charged output capacitor.

Security

Controlling V_{pp} provides the benefit of system hardware security. Beyond this, you can design for even higher security levels. The first level could be the design of a simple software password routine that would only turn on V_{pp} when a correct password is given. Alternatively, you can provide a jumper to allow 12V to the part for a BIOS update and then return it when reprogramming is finished. The system should check this pin to see if the jumper was left in the programming position and remind the user to move it. Unless V_{pp} is at 12V, the flash memory contents cannot be changed and acts just like ROM. Disabling V_{pp} until voltages have stabilized provides additional power-up protection.

The Motorola component, MC34064, is an under-voltage sensing circuit that begins functioning when V_{CC} is above 1V. Between 1V and 4.6V, the RESET# output is active. This (or a system RESET#) clears the 74FC74, keeping V_{pp} off. Alternatively, if you use CMOS logic, you could make use of Intel's flash memory V_{CC} and V_{pp} lockout functions. While V_{CC} is below 2.5V, the Command Register is locked out. Since CMOS control logic is active at 2.0V, a 0.5V safety margin exists for control logic to settle down before the part becomes active. Program and erase attempts are inhibited with V_{pp} below 6.5V. For both CMOS and non-CMOS designs (i.e., control logic active at 2.0V), gate RP# with the power supply's "Power Good" signal or the MC34064's RESET# output (Figure 13). Until RP# transitions to V_{IH} , the part ignores all write attempts, regardless of power supply voltages and bus activity.

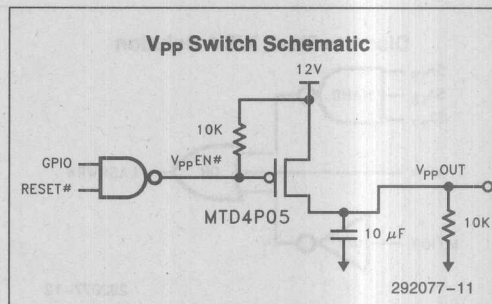


Figure 13. V_{pp} Switch Using MTD4P05

Using a MOSFET Switch

For laptops or palmtops, an always-active 12V may not provide acceptable power management. For these systems, a MOSFET switch will toggle 12V to the flash memory, minimizing current draw when not needed. Several DC switches exist, but there are a few issues to consider in your selection. Choose a switch with low "ON" resistance to keep the V_{pp} voltage within flash memory tolerances. The system 12V power supply must be specified to a tighter range to allow for any voltage drop through the switch. Allocate an I/O line (V_{pp} enable) to turn the switch on and off. To handle "warm RESETS", the V_{pp} enable must be gated with the system RESET# line. The Motorola MTD4P05 is one example of a surface-mount switch with low drain-source resistance. Assuming a 12V \pm 5% and -4% supply:

$$\begin{aligned} R_{DS} &= 0.6\Omega \\ I_{PP} &= 30 \text{ mA (Worst Case)} \\ \Delta V_{SWITCH} \text{ Drop} &= (30 \text{ mA} \times 0.6\Omega) = 0.02V, \\ &\leq (4\% \text{ of } V_{PP}) = 0.48V. \end{aligned}$$

Figure 13 shows a schematic of a V_{pp} switch design.

3.3 Modifying an Existing Motherboard

EPROM/ROM Designs

If you are modifying an existing motherboard design for a flash memory BIOS, there are a few things you should consider. First, check the logic design to determine if WR# is decoded and connected to the BIOS EPROM location. Typical motherboard logic designs do not allow writes to the EPROM locations and treat EPROM writes as invalid (e.g., ROMCS# not generated with MEMWR#). This is overcome by generating the BIOS location's WR# externally by either adding the necessary discrete logic or adding a 3-to-8 decoder (see Figure 14 for an example). In either case, tap into the M/IO# and WR# control lines and configure the decoder to provide a logic low for the M "AND" WR# "AND" BIOS address condition.

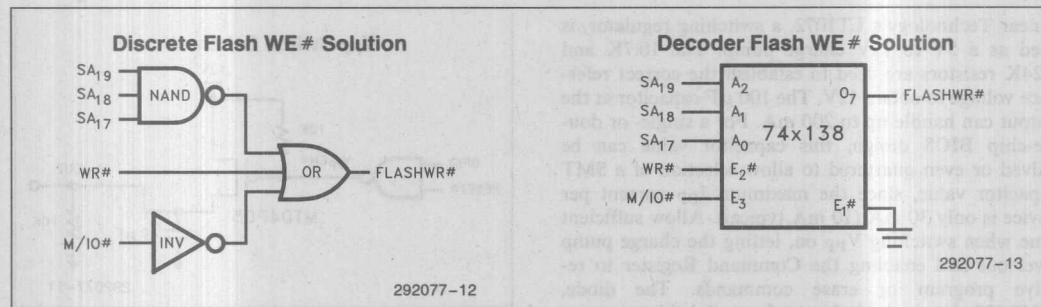


Figure 14. Discrete and Single-Chip Decoder WE# Solutions

Secondly, check to see if the BIOS code transceiver or buffer for the EPROM location works in both directions. The transceiver may need a special BIOS call to unlock it in the “write” direction, or you may have to reprogram the logic for that portion of your board. If your chip set data buffer works only in one direction, a transceiver and direction logic must be added to the CPU bus to pass data to and from flash memory.

Your system must also be capable of routing 12V to the BIOS socket for program and erase. Control RP# with system RESET# and POWERGOOD (see Section 3.0) and optionally provide capability for deep powerdown mode via an I/O line. Finally, address inversion or shift mechanisms outline in Section 3.1 can optionally be added for recovery capability with the 28F001BX.

28F010 Flash Memory Designs

If your design currently incorporates Intel's 28F010 flash memory, hardware upgrade to the 28F001BX is simple. Transceiver, BIOS write and V_{pp} requirements will have already been considered in the original design. Control RP# as described in Section 3.0. Finally, invert or shift the system addresses as in Section 3.1 if BIOS “ROM” access after shadowing to DRAM is anticipated.

3.4 In-System Write vs On-Board Programming

When devices are soldered directly to a printed circuit board, one of two sources control flash memory reprogramming:

1. the system's own processor, or
2. a PROM programmer connected to the board.

These options are called In-System Write (ISW) and On-Board Programming (OBP), respectively. Their respective benefits are discussed in detail in AP-316.

With ISW, the system drives the reprogramming process and generates V_{pp} locally. Under this scenario, the board manufacturer will initially program at least the boot block in a PROM programmer. This removes the need for circuitry on-board to unlock the boot block, guaranteeing boot code integrity throughout system life. A good design practice for ISW-type designs is to socket the first few flash BIOS prototypes. SMT-only designs can also socket using PLCC SMT sockets. Socketing enables the system designer to easily work out any bugs with in-system flash reprogramming by allowing the removal of a flash part for external reprogramming in a PROM programmer. Once ISW reprogramming is fully debugged, pre-programmed flash parts can be soldered directly to the circuit board without a socket. All flash memory components are exposed to a data-retention bake testing and checked for any data loss before shipping. It is extremely unlikely that data in a production flash device can be corrupted from heat by a production-run soldering application.

OBP uses an external board programmer to supply V_{pp} and V_{HH} and control the programming process. Certain design considerations must be evaluated prior to laying out the design. Some manufacturers using TSOP may also want to remove a handling step from the manufacturing process by providing the capability to program flash for the first time after being soldered directly onto the circuit board. OBP can accomplish this if the design is first laid out correctly to support OBP. External circuitry generates voltages needed to unlock and program/erase the boot block.

3.5 Ideas for Using Extra Adaptor Space

Laptop and palmtop systems may have adaptor space available in the system memory map since there typically isn't much room for add-in boards. Additionally, they may not use up the entire 128K of BIOS space due to their fixed feature set and limited upgrade capability. This extra memory space can hold ROM executable programs like Lotus 123, WordPerfect, Microsoft Works, etc. Using Intel's flash TSOPs, a small application cache can reduce a laptop's disk access and increase battery life.

Additionally, ROM-Executable DOS can be placed anywhere in adaptor space. For example, MS-DOS ROM Version 3.22 requires 62 KB of adaptor space today (this may change on subsequent revisions). One location for MS-DOS ROM Version 3.22 is directly un-

der the BIOS (again see Figure 8). Today's typical BIOS consumes 64 KB or less; consequently, both the BIOS and MS-DOS ROM Version 3.22 could reside in a single 28F001BX (128 Kbytes), yielding reduced chipcount. However, if power management code is added to the BIOS, system BIOS code could grow to 80 KB or more. Therefore, designs that include both power management and MS-DOS ROM Version 3.22 should consider using both a 28F001BX and 28F010 flash device (or two 28F001BX's). This leaves extra space for BIOS and MS-DOS ROM to grow in the design, while providing additional storage for the video BIOS.

4.0 SOFTWARE DESIGN CONSIDERATIONS

Intel's Flash Memory provides a cost-effective, updatable, nonvolatile code storage medium. The 28F001BX integrates the Quick-Pulse Programming and Quick-Erase algorithms of prior Intel Flash Memories on-chip, using the Command Register, Status Register and Write State Machine (WSM). On-chip integration dramatically reduces system overhead, simplifies system software creation and debug and provides SRAM-like timings to the Command and Status Registers. WSM operation, internal program/erase verify and V_{pp} high voltage presence are monitored and reported via appropriate Status Register bits. Table 1 lists the 28F001BX command set, while Table 2 details the Status Register bits and their meanings.

4

Table 1. 28F001BX Command Definitions

Command	Bus Cycles Req'd	Notes	First Bus Cycle			Second Bus Cycle		
			Operation	Address	Data	Operation	Address	Data
Read Array/Reset	1		Write	X	FFH			
Intelligent Identifier	3	1, 2, 3	Write	X	90H	Read	IA	IID
Read Status Register	2	2	Write	X	70H	Read	X	SRD
Clear Status Register	1		Write	X	50H			
Erase Setup/Erase Confirm	2	1	Write	BA	20H	Write	BA	D0H
Erase Suspend/Erase Resume	2		Write	X	B0H	Write	X	D0H
Program Setup/Program	2	1, 2	Write	PA	40H	Write	PA	PD

NOTES:

1. IA = Identifier Address; 00H for manufacturer code, 01H for device code.
BA = Address within the block being erased.
PA = Address of memory location to be programmed.
2. SRD = Data read from Status Register. See Table 2 for a description of Status Register bits.
PD = Data to be programmed at location PA. Data is latched on the rising edge of WE#.
IID = Data read from intelligent identifier.
3. Following the intelligent identifier command, two read operations access the manufacturer and device codes.
4. Commands other than shown above are reserved by Intel for future device implementations and should not be used.

Table 2. 28F001BX Status Register Definitions

WSMS	ESS	ES	PS	VPPS	R	R	R
7	6	5	4	3	2	1	0
<p>SR.7 = WRITE STATE MACHINE STATUS 1 = Ready 0 = Busy</p> <p>SR.6 = ERASE SUSPEND STATUS 1 = Erase Suspended 0 = Erase In Progress/Completed</p> <p>SR.5 = ERASE STATUS 1 = Error In Block Erase 0 = Successful Block Erase</p> <p>SR.4 = PROGRAM STATUS 1 = Error In Byte Program 0 = Successful Byte Program</p> <p>SR.3 = V_{PP} STATUS 1 = V_{PP} Low Detect; Operation Abort 0 = V_{PP} OK</p> <p>SR.2–SR.0 = RESERVED FOR FUTURE ENHANCEMENTS These bits are reserved for future use and should be masked out when polling the Status Register.</p>							
<p>NOTES:</p> <p>The Write State Machines Status Bit must first be checked to determine program or erase completion, before the Program or Erase Status bits are checked for success.</p> <p>If the Program AND Erase Status bits are set to 1's during an erase attempt, an improper command sequence was entered. Attempt the operation again.</p> <p>If V_{PP} low status is detected, the Status Register must be cleared before another program or erase operation is attempted.</p> <p>The V_{PP} Status bit, unlike an A/D converter, does not provide continuous indication of V_{PP} level. The WSM interrogates the V_{PP} level only after the program or erase command sequences have been entered and informs the system if V_{PP} has not been switched on. The V_{PP} Status bit is not guaranteed to report accurate feedback between V_{PPL} and V_{PPH}.</p>							

The WSM on-chip oscillator internally times the program/erase algorithms, making software timers unnecessary. Block precondition is also controlled by the WSM as part of the erase algorithm. Block data programming to "0's" before erasing is no longer needed.

Intel's high quality design, manufacturing and testing result in outstanding reliability and performance throughout device life. Although Program Status and Erase Status bits are provided for Status Register completeness, errors will probably not be encountered, if proper V_{PP} levels and software sequences are implemented.

Intel offers standard software drivers, written in "C", to assist software engineers implementing 28F001BX reprogramming for update utilities. These high-level routines, found in Appendix A, are adaptable to a wide range of μ P and μ C platforms and system architectures.

Covered in this section are the major software steps for a flash BIOS update utility:

- Update software for a modified system
- Pseudo-Code overview

- Initializing the system
- Code loader routine
- Flash re-programming
- Recovery routines
- Power management

4.1 Update Software for a Modified System

The design example of Section 3.0 assumes BIOS shadowing for BIOS code execution while allowing BIOS writes to the flash socket. Many systems provide a register which enables BIOS writes and reads. Some systems may not allow BIOS reads from RAM while performing BIOS writes to the flash socket, or vice versa. The reasons may be simple; no shadow RAM exists in the system (8088 or 8086 systems), or system logic treats "ROM writes" as an invalid operation. In these cases, perform all your required BIOS calls before you erase and program the flash memory. But keep in mind, to update the user on the progress of flash programming and indicate when programming is finished, you should add some basic screen or speaker "beep" routines to your update utility.

4.2 Pseudo-Code Overview

The following pseudo-code for an update utility provides a brief description of the process of updating a BIOS in-situ. It is based on software developed by a customer for a PC platform with BIOS update capability. This Intel386-33 MHz system uses the 28F001BX for BIOS storage. Modify the flowchart below, if needed, for your particular chipset and hardware environment.

Pseudo-Code for Flash Update Routine

Initialize system (set up user screen, check battery power, check device ID)

Get BIOS file options (from floppy or modem)

If no file present

Send error message to insert BIOS update floppy, or press ESC to exit

Display BIOS update files, prompt user for choice and load to memory

If file invalid,

Prompt for file or exit

Inform user what is about to happen, with option to continue or exit

If user continues, inform them to not turn off the power or soft-reboot (CNTL-ALT-DEL)

Erase 28F001BX main/parameter blocks

If system interrupt occurs

Suspend erase if flash memory access is required

Resume Erase

Write file[s] into flash memory

Indicate to user that flash reprogramming is over

Reboot the system

4.3 Initializing the System

Checking Power

If your application is a laptop or palmtop computer, first check the battery to make sure there is enough power to do the update. If not, inform the user to recharge the system before continuing the update and exit the update program. This ensures that the system won't stop in the middle of an update. Next, initialize access to flash for reads and writes, then try reading the device ID through the Command Register. Checking the device ID before programming or erasing helps determine

if reads and writes work correctly and that the flash memory in the system matches your code before starting to reprogram the part. The manufacturer ID for Intel flash memories is **89H** (10001001), located at device address 00000H. Device IDs are located at address 00001H; the ID for the 28F001BX-T is **94H** (10010100), and the 28F001BX-B device ID is **95H** (10010101). These device addresses, in the DOS memory map, correspond to system addresses E0000H (mfg. ID) and E0001H (device ID). If A₁₆ inversion is used as described in Section 3.1, system addresses for mfg. ID and device ID under normal operation are F0000H and F0001H.

NOTE:

During the initialization, you can also perform a scan of the adaptor space to ascertain if there is more flash in the system. Other Intel Flash Memories share common manufacturer IDs but have unique device IDs, listed below:

Device	Device ID (Hex)	Device ID (Binary)
28F256A	B9H	10111001
28F512	B8H	10111000
28F010	B4H	10110100
28F020	BDH	10111101
28F001BX-T	94H	10010100
28F001BX-B	95H	10010101

4.4 Code Loader Routine

The update utility described in the previous section provides an optional mouse-driven color graphical user interface (GUI) and allows not only BIOS update to the main block but also update of the parameter blocks, and copy/compare of block data to a DOS file. These types of features convey to the end user the ease and simplicity of performing a BIOS update. For example, the main block update utility lists all possible BIOS files in the selected drive and directory, and prompts the user for the desired file. System OEMs may want to encode a specific BIOS file name into the generic loader utility ".COM" or ".EXE" file. This allows automatic reading of the new BIOS file into a program buffer, bypassing the user prompt.

Once the file is loaded into RAM, the routine informs the user of the impending BIOS update and provides the option to exit if desired. If continued, it warns the user to not turn off power or reboot during the BIOS update procedure. It then erases and reprograms the main block with new BIOS data, notifies the user of successful update and reboots.

4.5 Flash Reprogramming Routines

On-Chip Erase Algorithm

The 28F001BX system erase algorithm is shown in Figure 15. Note that the actual device erase algorithm (Quick-Erase) is controlled internally, including all timing and block preconditioning. This provides the same high level of reliability proven on Intel's ETOX II technology, while reducing system debug efforts. Erase progress is reported to system software thru specific Status Register bits. The 28F001BX erases all bits of a block in parallel. Minimum and typical erase times for each block are listed below:

Block	Minimum Time (Sec)	Typical Time (Sec)
Parameter (ea.)	1.3	2.1
Main	3.0	3.8
Boot	1.3	2.1

The actual erase time depends on the V_{pp} voltage level (11.4V–12.6V), temperature and the number of erase cycles already completed on the part. System software must comprehend adequate time for V_{pp} , after enabled, to ramp to 12V before erase is attempted. Capacitors on the V_{pp} bus, in addition to the intrinsic pump nature of many 12V solutions, cause an RC ramp. Systems that direct-wire 12V need not worry about this delay.

Erase Suspend/Resume

Erase suspend gives the user the ability, while erasing a block of the 28F001BX, to read data or execute code from another block. This capability, in conjunction with the minimal system overhead provided by the WSM, makes disabling of interrupts during block erase unnecessary. Once given the erase suspend command, the WSM halts, reports suspend status to the Status Register and allows array reads. When issued erase resume, it proceeds at the point where it was suspended. Figure 16 details the system code flowchart that suspends and resumes erase.

The update utility described in the previous section provides an optional menu-driven option function used to interface (GUI) and allows not only BIOS updates to the main block but also updates of the parameter blocks. These and copy compare of block data to a DOS file. These types of features convey to the user that the user and simplicity of performing a BIOS update. For example, the main block update utility that all provide BIOS this is the selected drive and directory, and prompts the user for the desired file. System (DOS) may want to encode a specific BIOS file name into the generic update utility, ".COM" or ".EXE". The user selects the name of the new BIOS file into a program buffer, replacing the user prompt.

Once the file is loaded into RAM, the routine informs the user of the impending BIOS update and provides the option to exit if desired. If continued, it warns the user to not turn off power or remove the BIOS update procedure. It then checks and compares the main block with new BIOS data, updates the rest of successful update and reports.

4.3 Initializing the System

Checking Power

If your application is a laptop or battery computer, first check the battery to make sure there is enough power to do the update. If not, inform the user to re-charge the system before continuing the update and exit the update program. This ensures that the system won't stop in the middle of an update. Next, initialize access to flash for reading and writing, then by reading the device ID through the Command Register. Checking the device ID is done programming or creating data determine

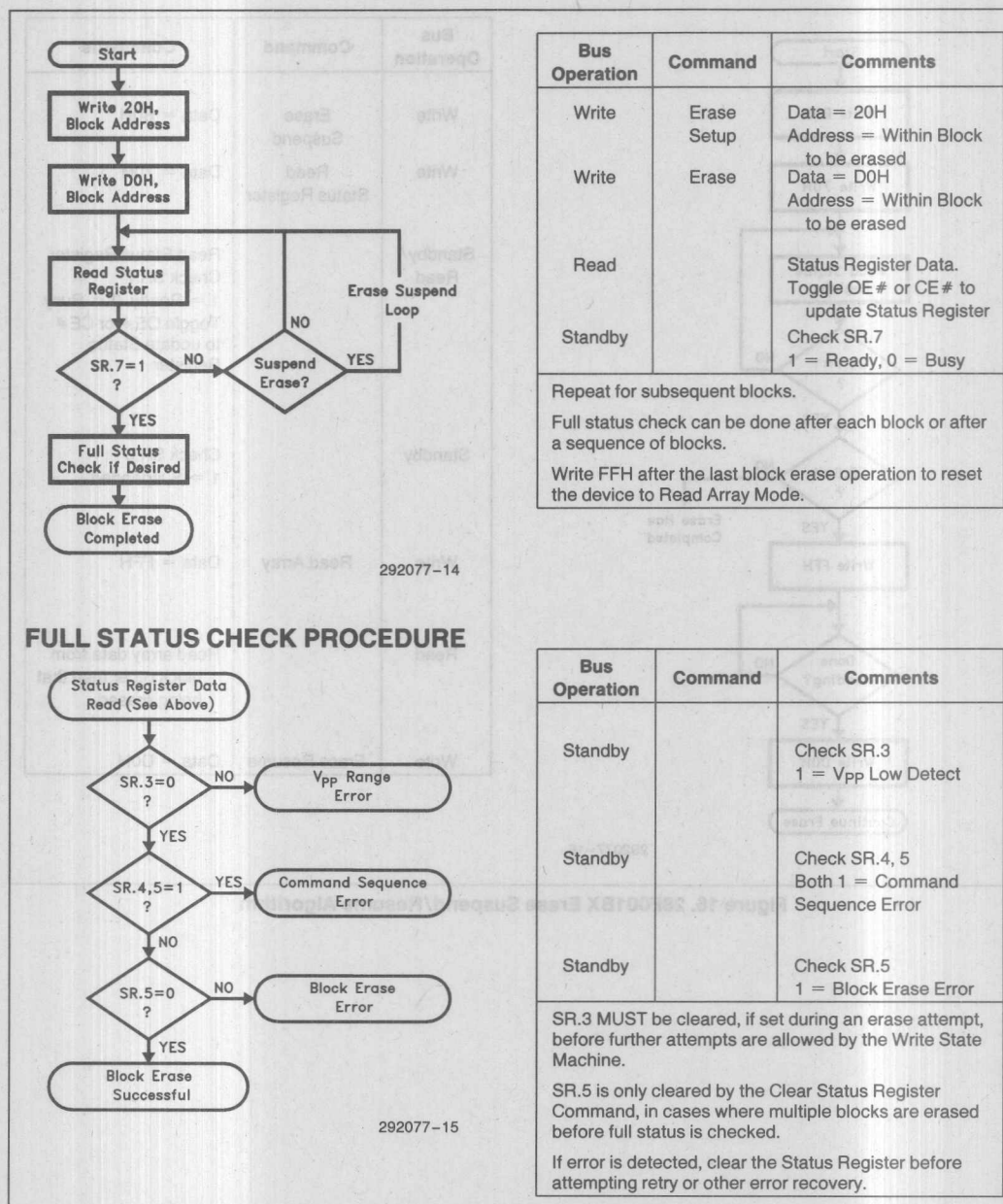


Figure 15. 28F001BX Block Erase Algorithm

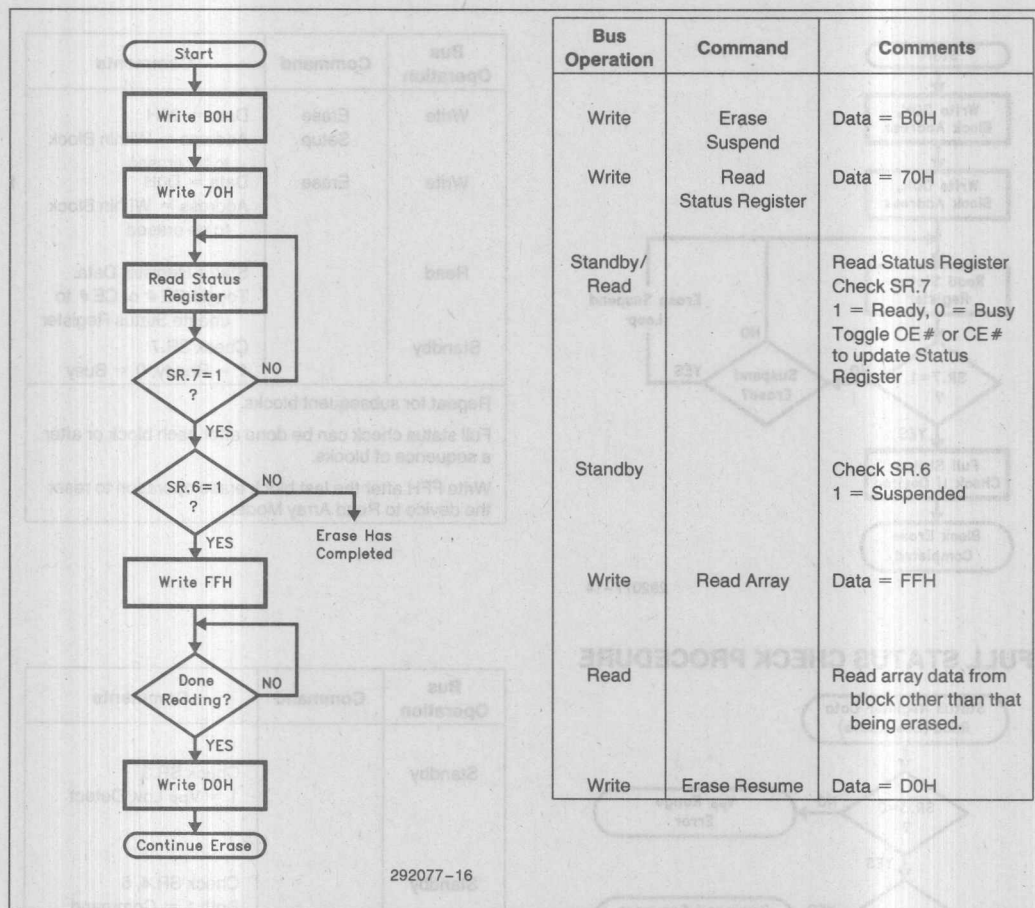
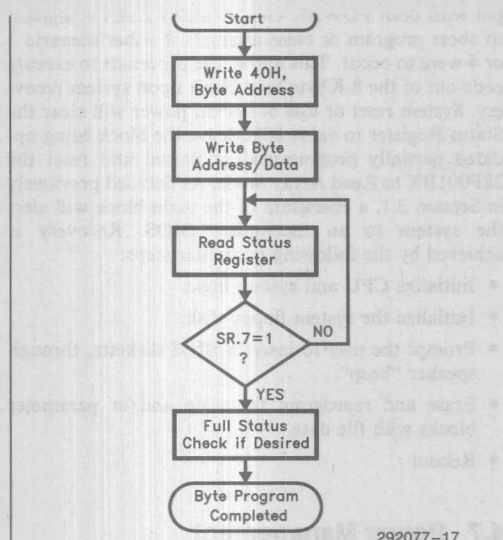
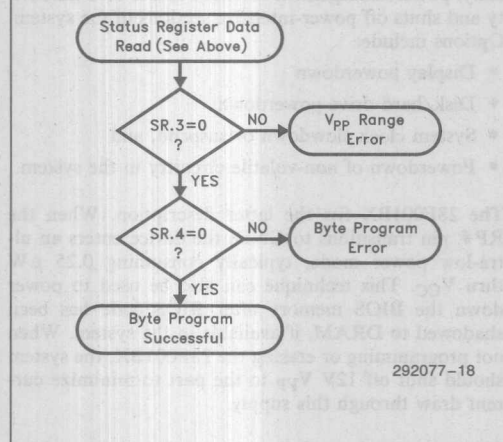


Figure 16. 28F001BX Erase Suspend/Resume Algorithm



292077-17

FULL STATUS CHECK PROCEDURE



292077-18

Figure 17. 28F001BX Byte Programming Algorithm

Bus Operation	Command	Comments
Write	Program Setup	Data = 40H Address = Byte to be programmed
Write	Program	Data to be Programmed Address = Byte to be programmed
Read		Status Register Data. Toggle OE# or CE# to update Status Register
Standby		Check SR.7 1 = Ready, 0 = Busy

Repeat for subsequent bytes.

Full status check can be done after each byte or after a sequence of bytes.

Write FFH after the last byte programming operation to reset the device to Read Array Mode.

Bus Operation	Command	Comments
Standby		Check SR.3 1 = Vpp Low Detect
Standby		Check SR.4 1 = Byte Program Error

SR.3 MUST be cleared, if set during a program attempt, before further attempts are allowed by the Write State Machine.

SR.4 is only cleared by the Clear Status Register Command, in cases where multiple bytes are programmed before full status is checked.

If error is detected, clear the Status Register before attempting retry or other error recovery.

On-Chip Programming Algorithm

As with 28F001BX erase, the Intel flash Quick-Pulse algorithm is internally controlled by the WSM. Figure 17 shows a system software flowchart for the Command Register/Status Register interface. Minimum and typical programming times (per byte) are 15 μ s and 18 μ s, respectively. Actual time varies with V_{pp} , temperature and cumulative programming cycles on the device. Ensure that stable 12V is applied to the device before attempting byte programming.

Full Status Checks

After polling the Status Register and determining that the WSM is again READY, system software should further analyze the Status Register to ensure that program or erase has successfully completed. The WSM will return to READY status after program or erase command sequences under any of the following conditions:

- Program/erase completed successfully,
- V_{pp} transition below specification during the program/erase attempt,
- Improper sequence of erase setup/confirm commands to the WSM, or
- Inability to erase the specified block, or program the desired byte.

Figures 15 and 17 detail the additional Status Register data analysis to ensure that program or erase have successfully occurred.

4.6 Recovery Routine Overview

Unsuccessful BIOS update can occur for any of the reasons listed below:

1. V_{pp} transitions out of specified tolerance during program or erase.
2. Incorrect code in the update BIOS disk file, or damaged BIOS disk.
3. Loss of system power during program or erase.
4. System reset (such as reboot) during program or erase.

The Status Register, through bit 3, reports V_{PPH} loss to system software. The BIOS update utility can detect scenario 1 and recover by simply re-attempting block update.

A checksum of update BIOS code after copy from disk to RAM, before flash erase and reprogram, will eliminate error caused by scenario 2.

PC motherboard logic should gate the 28F001BX RP# pin with both POWER GOOD and RESET# signals, to abort program or erase attempts if either scenario 3 or 4 were to occur. This allows the processor to execute code out of the 8 Kbyte boot block upon system recovery. System reset or loss of system power will clear the Status Register to value 80H, leave the block being updated partially programmed or erased and reset the 28F001BX to Read Array mode. As detailed previously in Section 3.1, a checksum of the main block will alert the system to an incomplete BIOS. Recovery is achieved by the following or similar steps:

- Initialize CPU and system logic.
- Initialize the system floppy disk.
- Prompt the user to insert a BIOS diskette, through speaker "beep".
- Erase and reprogram the main and/or parameter blocks with file data.
- Reboot

4.7 Power Management

Battery-powered PCs incorporate a variety of techniques to prolong system life between recharges. Typically, power management software senses user inactivity and shuts off power-intensive sections of the system. Options include:

- Display powerdown
- Disk/hard drive powerdown
- System clock slowdown or suspend, and
- Powerdown of non-volatile circuitry in the system.

The 28F001BX fits the latter description. When the RP# pin transitions to GND, the device enters an ultra-low power mode, typically consuming 0.25 μ W thru V_{CC} . This technique can also be used to power down the BIOS memory after BIOS code has been shadowed to DRAM, if available in the system. When not programming or erasing the 28F001BX, the system should shut off 12V V_{pp} to the part to minimize current draw through this supply.

User inactivity is typically detected if the keyboard has not been used, or the disk drive has not been accessed, for a predetermined interval (this is often user-programmable). Power management software must ensure that a BIOS update is not occurring, before powering down the 28F001BX, to prevent incomplete update.

For more information on power management techniques, consult datasheets and application notes on the Intel386SL microprocessor superiset.

5.0 SUMMARY

5.1 Traditional BIOS Storage and Disadvantages

Traditional BIOS storage has been in EPROM, which offers nonvolatility and factory programming capability. In earlier PCs, the BIOS code was fairly simple (relative to today's software) and updates were infrequent, so EPROMs or ROMs were an acceptable BIOS storage medium. Today's systems are much more sophisticated, with many designs supporting the Intel i386/i486™ microprocessors and new bus architectures like MCA and EISA for the first time. These new buses allow peripherals to take control of the system bus . . . it is difficult to guess what new system configurations might emerge. Therefore, the potential for a change in the BIOS code is much greater and the frequency of change is likely to increase.

A system designer may use EPROMs for BIOS storage to reduce initial system (component) costs, but the long-term update cost is much more than the difference between EPROM and flash memory components. A major manufacturer of PCs has estimated that a service call for a BIOS update with EPROMs can cost upwards of \$300.00 for ONE update at ONE site. EPROMs are also susceptible to bent leads during insertion by the technician, or more likely, the end user. Service is becoming a key differentiator between the multitudes of PC makers. Reducing the number of times a PC has to be opened for any reason and providing improved service increases customer confidence and promotes a reliable image.

5.2 Advantages of an Updatable BIOS

Using flash memory for BIOS storage provides a flexible code medium that allows the BIOS code to adapt to changing hardware and software conditions. BIOS updates in flash are inexpensive, via a floppy disk or modem. They remove EPROM inventories, reduce packaging requirements, reduce total postage costs and eliminate service cost for BIOS code updates by removing the need for a technician to do the update. A company that supports multiple OEMs can improve version management control by using a flash BIOS and floppies or a BBS for updates. An additional benefit is that not only the BIOS, but DOS itself can be stored in the same flash memory device.

5.3 Advantages of Adding DOS in FLASH

Once the requirements for flash memory BIOS are met, the capability is also in place for adding DOS in FLASH. Why put DOS in FLASH? For laptop and

palmtop PCs, battery longevity is of paramount concern, followed closely by weight and increasing user RAM (640 KB) space. Extra user RAM is needed for applications that require more than the typical 570 Kbytes (640 KB–70 KB) available with disk-based DOS. Digital Research Incorporated and Microsoft both make "DOS-in-ROM" products that address these needs. MS-DOS ROM version 3.22 is an example.

Microsoft's MS-DOS ROM Version 3.22 is a full-function version of MS-DOS 3.2. It features instant-on and employs only 15 KB of the 640 KB DOS RAM user space, leaving the rest for applications. Since MS-DOS ROM Version 3.22 loads from adaptor space, both disk access and DOS loadtime are reduced. For laptops, anything that can reduce disk access equates to battery longevity. Laptops can reduce weight by using MS-DOS ROM Version 3.22 and replacing the floppy drive with an IC card. Adding MS-DOS ROM Version to desktops also liberates additional user RAM for the same above reasons, but may not be optimal for high speed 32-bit systems.

All future versions of MS-DOS will be supported with equivalent versions of MS-DOS ROM. See Appendix B for more information.

5.4 Advantages of Adding 1 MB–4 MB of Resident Code Storage

There is a growing need for systems to be able to provide a small suite of bundled applications. Benefits to the user are faster application execution thru reduced hard or floppy disk access, no power used to store the resident code, and instant-on. No time is wasted transferring data over a disk I/O interface. The code is instead loaded to RAM with a simple memory copy function or procedure. In some cases, code is directly executed by the processor. Tandy's Deskmate is an example of such a system. Future versions of Deskmate-like user interfaces could easily be made flash-updatable. SRAM is too expensive and requires power to just store files. Furthermore, battery backup is not a reliable means of achieving nonvolatility. Intel's Flash Memory can provide user configurability for 1 MB–4 MB of code storage for just 2x–3x the cost of EPROMs and less than half the cost of SRAM. Applications such as Lotus 123, WordPerfect and Microsoft Works also come in either a direct-execute "ROM" version or a load-from-ROM format. Many other ROM application software packages are in development, servicing the successful and growing needs of the laptop/palmtop computers. Therefore, if an application can be stored or runs from ROM, it can be stored and run from flash. As software packages are periodically updated, flash memory provides the capability of updating these "ROM" applications at little cost to the software vendor and with no system disassembly required.

APPENDIX A SOFTWARE ROUTINES

```

/*****
/* Copyright Intel Corporation, 1991
/* Brian Dipert, Intel Corporation, July 14, 1991, Revision 1.4
/* The following drivers control the Command and Status Registers of
/* the 28F001BX Flash Memory to drive byte program, block erase, Status
/* Register read and clear and array read algorithms.
/* Sample Vpp and RP# control blocks are also included.
/* The functions listed below are included:
/* erasbgn(): Begins block erasure
/* erassusp(): Suspends erase to allow reading data from a block of the
/* 28F001BX other than that being erased
/* erasres(): Resumes erase if suspended
/* end(): Polls the Write State Machine to determine if block erase or
/* byte program have completed
/* eraschk(): Executes full status check after erase completion
/* probgn(): Begins byte programming
/* progchk(): Executes full status check after byte program completion
/* idread(): Reads and returns the manufacturer and device IDs of the
/* target 28F001BX
/* statrd(): Reads and returns the contents of the Status Register
/* statclr(): Clears the Status Register
/* rdmode(): Puts the 28F001BX in Read Array mode
/* rdbyte(): Reads and returns a specified byte from the target 28F001BX
/* vppup(): Enables high voltage Vpph
/* vppdown(): Disables Vpph
/* pwdon(): Ramps the RP# pin to high voltage Vhh, enabling boot block
/* program/erase
/* pwdoft(): Disables high voltage Vhh on RP#, disabling program
/* and erase of boot block
/*
/* Addresses are transferred to functions as pointers to far bytes (ie long
/* integers). An alternate approach is to create a global array the size of the
/* 28F001BX and locate "over" the 28F001BX in the system memory map. Accessing
/* specific locations of the 28F001BX is then accomplished by passing the chosen
/* function an offset from the array base versus a specific address. Different
/* microprocessor architectures will require different array definitions; ie for
/* the Intel architecture, define it as "byte boot [2][10000]" and pass each
/* function TWO offsets to access a specific location. MCS-51 architectures
/* are limited to "byte boot[10000]"; alternate approaches such as writing to
/* control bits will be required to access the full flash array
/*
/* To create a far pointer, a function such as MK_FP() can be used, given
/* a segment and offset in the Intel architecture. I use Turbo-C; see your
/* compiler reference manual for additional information.
*****/

```



```

/*****
/* Revision History: Rev 1.4
/*
/* Changes from 1.0 to 1.1: Added typedef for "byte" to accurately reflect
/* this x8 device. Altered variable definitions accordingly. Combined
/* functions progend() and erasend() into function end().
/*
/* Changes from 1.1 to 1.2: Added this revision history block. Added above
/* comments on alternate addressing methods.
/*
/* Changes from 1.2 to 1.3: Added pass/fail error return from idread(),
/* idread() at beginning of progbn() and erasbn(), pass/fail error
/* return from progbn() and erasbn().
/*
/* Changes from 1.3 to 1.4: Revised code to reflect simplified program and
/* erase algorithms. 28F001BX automatically transitions to Read Status Register
/* mode after program command sequence, erase command sequence and remains in
/* Read Status Register mode after Erase Suspend is issued. Address 0000H is no
/* longer required to read or clear the Status Register.
*****/

typedef unsigned char byte;

/*****
/* Function: Main
/* Description: Included only to omit errors when attempting to compile code.
/* The end customer would insert their main program here.
*****/

main()
{
}

/*****
/* Function: Erasbn
/* Description: Begins erase of a block.
/* Inputs: blkaddr: System address within the block to be erased
/* Outputs: None
/* Returns: 0 = Erase successfully initiated
/*          1 = Erase not initiated (ID check error)
/* Device Read Mode on Return: Status Register (ID if returns 1)
*****/

#define ERASETUP    0X20    /* Erase Setup command
#define ERASCONF    0XD0    /* Erase Confirm command

int erasbn(blkaddr)

byte far *blkaddr;          /* blkaddr is an address within the block to be
                             /*   erased

{
    if (idread()==1)          /* ID read error; device not powered up?
        return (1);
    *blkaddr = ERASETUP;      /* Write Erase Setup command to block address
    *blkaddr = ERASCONF;      /* Write Erase Confirm command to block address
    return (0);
}

```

```

/*****
/* Function: Erassusp
/* Description: Suspends block erase to read from another block
/* Inputs: None
/* Outputs: None
/* Returns: 0 = Erase suspended
/*          1 = Error; Write State Machine not busy (erase suspend not possible)
/* Device Read Mode on Return: Read Status Register
*****/

#define RDMASK 0X80 /* Mask to isolate the WSM Status bit of the
/* Status Register
#define WSMRDY 0X80 /* Status Register value after masking, signifying
/* that the WSM is no longer busy
#define SUSPMASK 0X40 /*Mask to isolate the Erase Suspend Status bit of the
/* Status Register
#define ESUSPYES 0X40 /* Status Register value after masking, signifying
/* that erase has been suspended
#define STATREAD 0X70 /* Read Status Register command
#define SYSADDR 0 /* This constant can be initialized to any address
/* within the memory map of the target 28F001BX
/* and is alterable depending on the system
/* architecture
#define SUSPCMD 0XB0 /* Erase Suspend command

int erassusp()

{
    byte far *stataddr; /* Pointer variable used to write commands to device

    stataddr = (byte far *)SYSADDR;
    *stataddr = SUSPCMD; /* Write Erase Suspend command to the device
    *stataddr = STATREAD; /* Write Read Status Register command to 28F001BX
    while ((*stataddr & RDMASK) != WSMRDY)
    {
        /* Will remain in while loop until bit 7 of the
        /* Status Register goes to 1, signifying that the
        /* WSM is no longer busy
    }
    if ((*stataddr & SUSPMASK) == ESUSPYES)
        return(0); /* Erase is suspended ... return code "0"
    return(1); /* Erase has already completed; suspend not possible.
    /* Error code "1"
}

```

```

/*****
/* Function: Erasres
/* Description: Resumes block erase previously suspended
/* Inputs: None
/* Outputs: None
/* Returns: 0 = Erase resumed
/*          1 = Error; Erase not suspended when function called
/* Device Read Mode on Return: Status Register
*****/

#define RDYMASK    0X80    /* Mask to isolate the WSM Status bit of the
/*                          Status Register
#define WSMRDY     0X80    /* Status Register value after masking, signifying
/*                          that the WSM is no longer busy
#define SUSPMASK   0X40    /* Mask to isolate the Erase Suspend Status bit
/*                          of the Status Register
#define ESUSPYES   0X40    /* Status Register value after masking, signifying
/*                          that erase has been suspended
#define STATREAD   0X70    /* Read Status Register Command
#define SYSADDR    0       /* This constant can be initialized to any
/*                          address within the memory map of the target
/*                          28F001BX and is alterable depending on the
/*                          system architecture
#define RESUMCMD    0XD0    /* Erase Resume Command

int erasres()
{
    byte far *stataddr;    /* Pointer variable used to write commands to device */

    stataddr = (byte far *)SYSADDR,
    *stataddr = STATREAD;    /* Write Read Status Register command to 28F001BX */
    if ((*stataddr & SUSPMASK) != ESUSPYES)
        return (1);    /* Erase not suspended. Error code "1" */
    *stataddr = RESUMCMD;    /* Write Erase Resume command to the device */
    while ((*stataddr & SUSPMASK) == ESUSPYES)
        ;    /* Will remain in while loop until bit 6 of the
        /*      Status Register goes to 0, signifying
        /*      erase resumption
    while ((*stataddr & RDYMASK) == WSMRDY)
        ;    /* Will remain in while loop until bit 7 of the
        /*      Status Register goes to 0, signifying
        /*      that the WSM is once again busy

    return (0);
}

```

```

/*****
/* Function: End
/* Description: Checks to see if the WSM is busy
/* (is program/erase completed?)
/* Inputs: None
/* Outputs: statdata: Status Register data read from device
/* Returns: 0 = Program/Erase completed
/* 1 = Program/Erase still in progress
/* Device Read Mode on Return: Status Register
*****/

#define RDYMASK    0X80    /* Mask to isolate the WSM Status bit of the
/* Status Register
#define WSMRDY     0X80    /* Status Register value after masking, signifying
/* that the WSM is no longer busy
#define STATREAD   0X70    /* Read Status Register command
#define SYSADDR    0       /* This constant can be initialized to any
/* address within the memory map of the target
/* 28F001BX and is alterable depending on the
/* system architecture

int end (statdata)

byte *statdata;          /* Allows Status Register data to be passed back
/* to the main program for further analysis

{
    byte far *stataddr;   /* Pointer variable used to write commands to
/* device
    stataddr = (byte far*)SYSADDR;
    *stataddr = STATREAD; /* Write Read Status Register command to 28F001BX
    if (((*statdata = *stataddr) & RDYMASK) != WSMRDY)
        return (1);      /* Program/erasure still in progress...code "1"
    return (0);           /* Program/erase attempt completed...code "0"
}

```



```

Description: Completes full status register check for erase (proper
/*      command sequence, Vpp low detect, erase success). This routine assumes
/*      that erase completion has already been checked in function end() and
/*      therefore does not check the WSM Status bit of the Status Register
/*      Inputs: statdata: Status Register data read in function end
/*      Outputs: None
/*      Returns: 0 = Erase completed successfully
/*              1 = Error; Vpp low detect
/*              2 = Error; Block erase error
/*              3 = Error; Improper command sequencing
/*      Device Read Mode on Return: Same as when entered
/*****

```

```

#define ESEQMASK    0X30 /* Mask to isolate the Erase and Program
/*      Status bits of the Status Register
#define ESEQFAIL    0X30 /* Status Register value after masking if erase
/*      command sequence error has been detected
#define ERRMSK      0X20 /* Mask to isolate the Erase Status bit of the
/*      Status Register
#define ERASERR      0X20 /* Status Register value after masking if erase error
/*      has been detected
#define VLOWMASK     0X08 /* Mask to isolate the Vpp Status bit of the Status
/*      Register
#define VPPLow      0X08 /* Status Register value after masking if Vpp low
/*      has been detected

```

```

int eraschk(statdata)

```

```

    byte statdata; /* Status Register data that has been already read
/*      from the 28F001BX in function end()

{
    if ((statdata & VLOWMASK) == VPPLow)
        return (1); /* Vpp low detect error, return code "1"
    if ((statdata & ERRMSK) == ERASERR)
        return (2); /* Block erase error detect, return code "2"
    if ((statdata & ESEQMASK) == ESEQFAIL)
        return (3); /* Erase command sequence error, return code "3"
    return (0); /* Block erase success, return code "0"
}

```

```

/*****
/* Function: Progbtn
/* Description: Begins byte program sequence
/* Inputs: pdata: Data to be programmed into the device
/*          paddr: Target address to be programmed
/* Outputs: None
/* Returns: 0 = Program successfully initiated
/*          1 = Program not initiated (ID check error)
/* Device Read Mode on Return: Status Register (ID if returns 1)
*****/

#define SETUPCMD    0X40    /*Program Setup command

int progbtn (pdata,paddr)

byte pdata;          /* Data to be programmed into the 28F001BX
byte far *paddr;      /* paddr is the destination address for the data
                      /* to be programmed

{
    if (idread() == 1)    /* Device ID read error...powered up?
        return (1);
    *paddr = SETUPCMD;    /* Write Program Setup command and
                          /* destination address
    *paddr = pdata;       /* Write program data
                          /* and destination address

    return (0);
}

```

```

/*****
/* Function: Progchk
/* Description: Completes full Status Register check for byte program (Vpp low
/* detect, programming success). This routine assumes that byte program
/* completion has already been checked in function end() and
/* therefore does not check the WSM Status bit of the Status Register
/* Inputs: statdata: Status Register data read in function end
/* Outputs: None
/* Returns: 0 = Byte programming completed successfully
/*          1 = Error; Vpp low detect
/*          2 = Error; Byte program error
/* Device Read Mode on Return: Status Register
*****/

#define PERRMSK 0X10 /* Mask to isolate the Program Status bit of the
/* Status Register
#define PROGERR 0X10 /* Status Register value after masking if program
/* error has been detected
#define VLOWMASK 0X08 /* Mask to isolate the Vpp Status bit of the Status
/* Register
#define VPPLow 0X08 /* Status Register value after masking if Vpp low
/* has been detected

int progchk (statdata)

byte statdata; /* Status Register data that has been already read
/* from the 28F001EX in function end()

{
    if ((statdata & VLOWMASK) == VPPLow)
        return (1); /* Vpp low detect error, return code "1"
    if ((statdata & PERRMSK) == PROGERR)
        return (2); /* Byte program error detect, return code "2"
    return (0); /* Byte/string program success, return code "0"
}

```

```

/*****
/* Function: Idread
/* Description: Reads the manufacturer and device IDs from the target 28F001BX
/* Inputs: None
/* Outputs: mfgrid: Returned manufacturer ID
/*           deviceid: Returned device ID
/* Returns: 0 = ID read correct
/*           1 = Wrong or no ID
/* Device Read Mode on Return: Intelligent Identifier
*****/

#define MFGRADDR      0      /* Address "0" for the target 28F001BX...
/*                               alterable depending on the system
/*                               architecture
#define DEVICADD      1      /* Address "1" for the target 28F001BX...
/*                               alterable depending on the system
/*                               architecture
#define IDRDCOMM      0X90    /* Intelligent Identifier command
#define INTELID       0X89    /* Manufacturer ID for Intel devices
#define DVCIDBT       0X94    /* Device ID for 28F001BX-T; change to 95H if
/*                               using 28F001BX-B!!!

int idread(mfgrid,deviceid)

byte *mfgrid;                /* The manufacturer ID read by this function, to
/*                               be transferred back to the calling
/*                               program
byte *deviceid;              /* The device ID read by this function, to be
/*                               transferred back to the calling function */

{
byte far *tempaddr;          /* Pointer address variable used to read IDs
tempaddr = (byte far*)MFGRADDR;
*tempaddr= IDRDCOMM;          /* Write intelligent identifier command to an
/*                               address within the 28F001BX memory map
/*                               (in this case, 00H)
*mfgrid = *tempaddr;          /* Read mfg ID, tempaddr still points at address "0"
tempaddr = (byte far*)DEVICADD; /* Point to address "1" for the device specific ID
*deviceid= *tempaddr;          /* Read device ID
if ((*mfgrid != INTELID)||(*deviceid != DVCIDBT))
return (1);                  /* ID read error; device powered up?
return (0);
}

```



```

/*****
/*  Function: Statrd
/*      Description: Returns contents of the target 28F001BX Status Register
/*      Inputs: None
/*      Outputs: statdata: Returned Status Register data
/*      Returns: Nothing
/*      Device Read Mode on Return: Status Register
*****/

#define STATREAD    0X70 /* Read Status Register command
#define SYSADDR    0 /* This constant can be initialized
                        /* to any address within the
                        /* memory map of the target 28F001BX
                        /* and is alterable depending on
                        /* the system architecture

int statrd(statdata)

byte *statdata; /* Allows Status Register data to
                /* be passed back to the calling program
                /* for further analysis

{
    byte far *stataddr; /* Pointer variable used to write
                        /* commands to device

    stataddr = (byte far*)SYSADDR;
    *stataddr = STATREAD; /* Write Read Status Register
                        /* command to 28F001BX

    *statdata = *stataddr;
    return;
}

```

```

/*****
/* Function: Statclr
/* Description: Clears the 28F001BX Status Register
/* Inputs: None
/* Outputs: None
/* Returns: Nothing
/* Device Read Mode on Return: Status Register
*****/

#define STATCLR    0X50    /* Clear Status Register command
#define SYSADDR    0       /* This constant can be initialized to any
                           /* address within the memory map of the target
                           /* 28F001BX and is alterable depending on
                           /* the system architecture

int statclr()
{
    byte far *stataddr;    /* Pointer variable used to write commands to
                           /* device

    stataddr = (byte far*)SYSADDR;
    *stataddr = STATCLR;    /* Write Clear Status Register command to
                           /* 28F001BX

    return;
}

/*****
/* Function: Rdmode
/* Description: Puts the target 28F001BX in Read Array Mode. This function
/* might be used, for example, to prepare the system for return to code
/* execution out of the flash memory after program or erase algorithms
/* have been executed off-chip
/* Inputs: None
/* Outputs: None
/* Returns: Nothing
/* Device Read Mode on Return: Array
*****/

#define RDARRAY    0XFF    /* Read Array command
#define SYSADDR    0       /* This constant can be initialized to any
                           /* address within the memory map of the target
                           /* 28F001BX and is alterable depending on
                           /* the system architecture

int rdmode()
{
    byte far *tempaddr;    /* Pointer variable used to write commands to
                           /* device

    tempaddr = (byte far*)SYSADDR;
    *tempaddr = RDARRAY;    /* Write Read Array command to 28F001BX

    return;
}

```

```

/*****
/* Function: Rdbyte
/* Description: Reads a byte of data from a specified address and
/* returns it to the calling program
/* Inputs: raddr: Target address To be read from
/* Outputs: rdata: Data at the specified address
/* Returns: Nothing
/* Device Read Mode on Return: Array
*****/

#define RDARRAY 0XFF /* Read array command */

int rdbyte (rdata,raddr)

byte *rdata; /* Returns data read from the device at
/* specified address
byte far *raddr; /* Raddr is the target address to be read from
{
*raddr = RDARRAY; /* Write read array command to an address within
/* the 28F001EX memory map (in this case the
/* target address)
*rdata = *raddr; /* Read from the specified address and store
return;
}

```

```

/*****
/*  Function: Vppup
/*  Description: Ramps the Vpp supply to the target 28F001BX to enable
/*  programming or erase. This routine can be tailored to the individual
/*  system architecture. For purposes of this example, I assumed that a
/*  system Control Register existed at system address 20000 hex.
/*  with the following definitions:
/*
/*  Bit 7: Vpph Control:      1 = Enabled
/*                           0 = Disabled
/*  Bit 6: PWD Control:       1 = PowerDown Enabled
/*                           0 = PowerDown Disabled
/*  Bits 5-0: Undefined
/*  Inputs: None
/*  Outputs: None
/*  Returns: Nothing
/*  Device Read Mode on Return: As existed before entering the function.
/*  Part is now ready for program or erase command sequence
*****/

#define VPPHIGH      0X80 /* Bit 7 = 1, Vpp elevated to Vpph */
#define SYSCADDR     0X20000 /* Assumed system Control Register Address */

int vppup()

{
    byte far *contaddr; /* Pointer variable used to write data */
                        /* to the System Control Register */

    contaddr = (byte far*) SYSCADDR;
    *contaddr = *contaddr | VPPHIGH; /* Read current Control Register data, */
                                    /* "OR" with constant to ramp Vpp */

    return;
}

```



```

/* ***** */
/* Function: Vppdown */
/* Description: Ramps down the Vpp supply to the target 28F001BX to */
/* disable programming/erase. See above for a description of the */
/* assumed system Control Register. */
/* Inputs: None */
/* Outputs: None */
/* Returns: Nothing */
/* Device Read Mode on Return: As existed before entering the function. Part */
/* now has high Vpp disabled. If program or erase was in progress when */
/* this function was called, it will complete unsuccessfully with Vpp low error */
/* in the Status Register. */
/* ***** */

#define VPPDWN      0X7F      /* Bit 7 = 0, Vpp lowered to Vppl */
#define SYSCADDR    0X20000 /* Assumed system Control Register Address */

int vppdown()
{
    byte far *contaddr; /* Pointer variable used to write data to the */
                        /* system Control Register */
    contaddr = (byte far*)SYSCADDR;

    *contaddr = *contaddr & VPPDWN;
                /* Read current Control Register data, "AND" with */
                /* constant to lower Vpp */

    return;
}

```

```

/*****
/* Function: Pwdon
/* Description: Toggles the 28F001EX RP# pin low to put the device in Deep
/* PowerDown mode. See above for a description of the assumed
/* system Control Register.
/* Inputs: None
/* Outputs: None
/* Returns: Nothing
/* Device Read Mode on Return: The part is powered down. If program or erase
/* was in progress when this function was called, it will abort with
/* resulting partially programmed or erased data. Recovery in the form of
/* repeat of program or erase will be required once the part
/* transitions out of powerdown, to initialize data to a known state.
*****/

#define PWD          0X40          /* Bit 6 = 1, RP# enable
#define SYSCADDR     0X20000      /* Assumed system Control Register Address

int pwdon()

{
    byte far *contaddr;          /* Pointer variable used to write data to the
                                /* system Control Register
    contaddr = (byte far*)SYSCADDR;
    *contaddr = *contaddr | PWD; /* Read current Control Register data, "OR" with
                                /* constant to enable Deep PowerDown
    return;
}

```

```

/*****
/* Function: Pwdoff
/* Description: Toggles the 28F001BX RP# pin high to transition the part
/* out of Deep PowerDown. See above for a description of the assumed system
/* Control Register.
/* Inputs: None
/* Outputs: None
/* Returns: Nothing
/* Device Read Mode on Return: Read Array mode. Low voltage is removed
/* from RP#. 28F001BX output pins will output valid data time tPHQV
/* after the RP# pin transitions high (reference the datasheet AC
/* Read Characteristics) assuming valid states on all other control
/* and power supply pins.
*****/

#define PWDOFF      OXBF      /* Bit 6 = 0, RP# disabled
#define SYSCADDR    OX20000   /* Assumed system Control Register Address

int pwdoff()
{
    byte far *contaddr;      /* Pointer variable used to write data to the
                             /* system Control Register
    contaddr = (byte far*)SYSCADDR;
    *contaddr = *contaddr & PWDOFF; /* Read current Control Register data, "AND" with
                             /* constant to disable Deep PowerDown
    return;
}

```

APPENDIX B MS-DOS ROM VERSION OVERVIEW

Technical Highlights

(Taken from Microsoft Product Overview)

RAM Economy

Because MS-DOS ROM Version executes from ROM, only 15 KB of system RAM space is required for MS-DOS. For a typical user, this will result in a savings of about 40 KB of RAM over disk-based MS-DOS. As a result of this savings, the user is able to run more programs and work with larger data files with the ROM Version than with disk-based MS-DOS. Instant-On MS-DOS ROM Version provides a significant reduction in "boot time", or the amount of time it takes from the completion of the power-on self test until a DOS prompt appears. With the ROM Version, this typically takes one second.

No End-User Installation

MS-DOS ROM Version is pre-installed by the OEM (original equipment manufacturer) in the system, thus freeing end users from the task of installing MS-DOS.

Adaptable to OEM Hardware Platforms

MS-DOS ROM Version is structured such that it allows the OEM to include a specific routine to determine which drive to boot from and any specific parameters if booting from the ROM drive. This makes it possible to easily port the ROM Version to a wide variety of hardware environments. MS-DOS ROM Version

is also positioned independent, in that it can reside anywhere in the "reserved" space (the area between 640 KB and 1 MB). This provides an additional Version to the specific requirements of the OEM's hardware platform.

ROM Economy

MS-DOS ROM Version occupies only 62 KB of ROM space, thus minimizing the amount of ROM that an OEM must include in the system. Three modules reside in the reserved space: COMMAND.COM, IO.SYS and the DOS Kernel. All three are position independent, so an OEM can decide where to place these modules in the reserved area.

National Language Support

Microsoft offers a full compliment of localized version of MS-DOS ROM Version, including Kanji and Chinese translations.

Ease of Development

As PCs become the engines for many embedded applications, manufacturers would like to develop new applications utilizing existing PC software tools. MS-DOS ROM allows manufacturers to take full advantage of these tools. For instance, a programmer can develop and debug an application onto a PC subsystem which may be embedded into a larger system. This benefit translates into a cost savings when developing a solution for vertical markets.

APPENDIX C BIOS VENDOR INFORMATION

American Megatrends Inc. (AMI)
1346 Oakbrook Drive, Suite 120
Norcross, GA 30093
(404) 263-8181

Award Software Inc.
130 Knowles Drive
Los Gatos, CA 95030
(408) 370-7979

Phoenix Technologies, LTD.
40 Airport Parkway
San Jose, CA 95110
(408) 452-6500

Systemsoft Corporation
313 Speen Street
Natick, MA 01760
(508) 651-0088

This list is intended for example only, and in no way represents all companies that support BIOS software. Since this industry develops many new solutions each year, Intel recommends that the designer contact the vendors for their latest products. Intel will continue to work with BIOS vendors to develop optimum solutions. Intel Corporation assumes no responsibility for circuitry or software other than circuitry embodied in Intel products. No software patent licenses are implied.

APPENDIX D

MICROPROCESSOR/MICROCONTROLLER COMPATIBILITY CHART

28F001BX-T	28F001BX-B
x86 Family	i960 KA/KB Microprocessor
i860™ Family	i960 SA/SB Microprocessor
i960 CA Microprocessor	MCS®-51 Family
	MCS®-96 Family

REVISION HISTORY

Number	Description
-005	Changed PWD# to RP# to match JEDEC naming conventions. Updated RP# control circuitry of Figure 7.

Extended Flash Bios Concepts For Portable Computers

4

SALIM FEDEL
SENIOR APPLICATIONS ENGINEER

October 1993

Extended Flash Bios Concepts for Portable Computers

CONTENTS	PAGE	CONTENTS	PAGE
1.0 INTRODUCTION	4-253	5.0 SOFTWARE DESIGN CONSIDERATIONS	4-262
2.0 PC BIOS TODAY AT THE 128 KBYTE CODE SIZE LIMIT	4-253	5.1 16 Kbyte Recovery Code	4-264
3.0 WHY BIOS CODE WILL GROW BEYOND 128 KBYTES	4-253	5.2 28F200BX Reprogramming	4-264
3.1 Advanced Power Management ..	4-255	5.3 Power Management	4-264
3.2 Optimizing New Portable Applications	4-255	6.0 DESIGNING A 3.3V SYSTEM	4-265
3.3 Putting Microsoft* MS-DOS 5.0 Operating System ROM Version into the BIOS Chip	4-255	6.1 Low Voltage Chips	4-265
3.4 Relocated Resident VGA Code: VIDEO BIOS	4-255	6.2 Power Savings and Improved Battery Life	4-265
4.0 HARDWARE DESIGN FOR A 256-KBYTE BIOS	4-256	7.0 CONCLUSIONS	4-265
4.1 Intel 28F200BX/002BX Boot Block Flash Memory Family	4-256	7.1 Benefits of Extended Flash BIOS	4-265
4.2 Extended Flash BIOS Design Example	4-257	REFERENCES	4-265
4.3 The ISA Sliding Window	4-262	APPENDIX A	4-266
4.4 The 28F200BX-T/28F002BX-T in the 1 Mbyte DOS Memory Map	4-262	APPENDIX B	4-267
		APPENDIX C	4-272

*Microsoft is a registered trademark of Microsoft Corporation.

1.0 INTRODUCTION

PC BIOS has been migrating to flash-based designs with the introduction of highly optimized flash memory architectures. The first phase of this shift in paradigm was from ROM/EPROM-based BIOS to Bulk Erase Flash memory-based BIOS to provide for in-system updatable BIOS and hence an easy update capability when BIOS changes are required.

The second phase improved the basic flash design to migrate towards boot block flash memory architecture with the Intel 28F001BX Flash Memory. This improvement enabled the implementation of additional features and provided a true design capability for portable PC BIOS.

The third phase of this paradigm shift now starting to evolve, deals with the need to grow beyond the traditional BIOS space limit of 128 Kbytes imposed by the original PC architecture to accommodate the advanced features of today's portable and desktop systems.

This application note describes in detail this third phase in BIOS hardware and software implementation. Specifically it will investigate why BIOS needs to grow beyond the 128 Kbyte code size. Then, a design example using the Intel 28F200BX Boot Block Flash Memory will be explained in terms of both hardware and software. Finally, low voltage PC BIOS designs incorporating 3.3V components are described.

2.0 PC BIOS TODAY AT THE 128 KBYTE CODE SIZE LIMIT

The Basic Input/Output System (BIOS) code is the lowest system level software which manages the interaction between all hardware components (CPU, Chip-Sets and I/O) with all software modules (Operating Systems and Applications Code). BIOS manages many functions in a PC, such as Power-On Self Test (POST), input vector creation, I/O services and system initialization. Therefore BIOS is the essential interface layer for full system functionality and compatibility.

The original PC architecture, developed in 1981, put restrictions on the size and mapping of the BIOS code which was then a very simple piece of software (on the order of 32 Kbytes for the original BIOS code). It was located at the top of the PC's (8088) memory map which at the time was a maximum of 1 Mbyte.

Then in 1984, the PC AT (80286) BIOS was expanded another 32 Kbytes for a total of 64 Kbytes. Subsequently, towards the late 1980s, more elaborate BIOS set-up utilities started to be an integral part of the BIOS code. In addition, personal computer manufacturers designed custom features into their BIOS code to offer more system flexibility. This increase in code complexity expanded the AT BIOS code another 64 Kbytes (for a total of 128 Kbytes) To occupy the total BIOS reserved space in the 1 Mbyte memory map.

In the DOS memory map, BIOS is mapped down from the top of the 1-Mbyte address space (F0000H to FFFFFH). Additional BIOS code space is available for future enhancements from E0000H to EFFFFH. The next 256 Kbytes in the DOS memory map are reserved for adapter space to accommodate add-in boards (for enhanced graphics cards for instance). Finally the remaining 640 Kbytes are reserved for the user to load his/her applications for execution. See Figure 1 for a graphical description.

3.0 WHY BIOS CODE WILL GROW BEYOND 128 KBYTES

As advances in computer design affect both desktop systems (with the addition of EISA, PCI and on-board SCSI capabilities) and portable systems (with the addition of advanced power management capability and I/O cards), the need for larger amounts of non-volatile memory space becomes evident.

A Notebook or a Palmtop computer design, for instance, may put the operating system, the system management code, set-up or utility programs into the non-volatile memory area to conserve precious RAM space for applications.

Additionally, Video BIOS can also be mapped into the Flash BIOS area.

Therefore, to implement advanced capabilities and provide new features (as described above) into powerful mobile computers, the 128 Kbyte BIOS code size limit had to be removed as it shall be explored in the next section.

The BIOS of today and the future must adapt to the new requirements of portable PC designs and take advantage of the new capabilities of low power PC chipsets and I/O devices to achieve the highest performance and longest battery life at the lowest system cost.

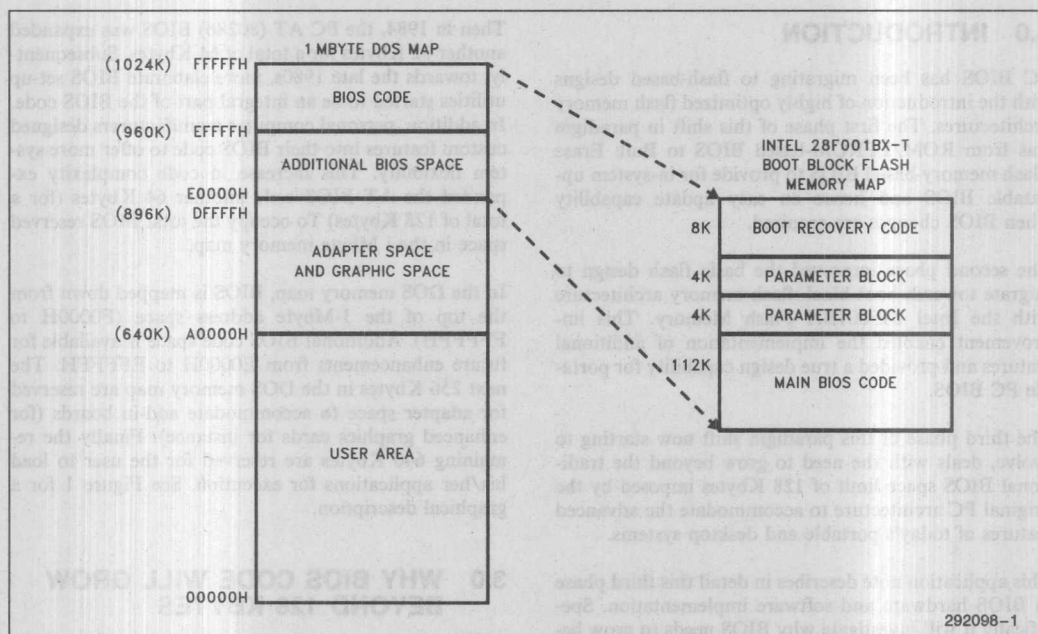


Figure 1. 128 Kbyte BIOS Code Segmentation in 1-Mbyte DOS Memory Map

BIOS capabilities) and portable systems (with the addition of advanced power management capability and I/O cards), the need for larger amounts of non-volatile memory space becomes evident.

A Notebook or a Tablet computer design, for instance, may put the operating system, the system management code, set-up or utility programs into the non-volatile memory area to conserve precious RAM space for applications.

Additionally, Video BIOS can also be mapped into the Flash BIOS area.

Therefore, to implement advanced capabilities and provide new features (as described above) new powerful mobile computers, the 128 Kbyte BIOS code size limit had to be removed as it shall be replaced in the next section.

The BIOS of today and the future must adjust to the new requirements of portable PC designs and the advantages of the new capabilities of low power PC chips and I/O devices to achieve the highest performance and longest battery life in the lowest system cost.

will be explained in terms of both hardware and software. Finally, low voltage PC BIOS designs incorporating 3.3V components are described.

2.0. PC BIOS TODAY AT THE 128 KBYTE CODE SIZE LIMIT

The Basic Input/Output System (BIOS) code is the lowest system level software which manages the interaction between all hardware components (CPU, Chip-set and I/O) with all software modules (Operating Systems and Applications Code). BIOS manages many functions in a PC, such as Power-On Self Test (POST), input vector redirection, I/O services and system initialization. Therefore, BIOS is the essential interface layer for full system functionality and compatibility.

The original PC architecture, developed in 1981, put restrictions on the size and mapping of the BIOS code which was then a very simple piece of software (on the order of 32 Kbytes for the original BIOS code). It was located at the top of the PC's (8088) memory map which at the time was a maximum of 1 Mbyte.

3.1 Advanced Power Management

High integration CPUs and chip-sets allow for the design of light, small form factor portable computers with long battery life.

BIOS is the ideal place for implementing the power management techniques pioneered with the Intel386SL Microprocessor Superset. BIOS software vendors have implemented APM code for the latest generation of Notebook PCs.

This added level of functionality imposed on the BIOS code increases the need for larger code space beyond the traditional 128 Kbyte BIOS implementations seen in today's portable systems.

APM code typically requires an additional 32 Kbyte of code space beyond the basic 64 Kbyte standard BIOS. Therefore, with the addition of APM, BIOS code grows to 96 Kbytes.

3.2 Optimizing New Portable Applications

With the implementation of PCMCIA cards for non-volatile file storage (with flash memory cards) and the ability to communicate over a telephone line (with modem cards), mobile computers have truly become powerful tools on the road.

The establishment of a common PC card standard for full system compatibility is described in the PCMCIA Standard release 2.0 (Personal Computer Memory Card International Association). Intel has developed a similar set of specifications fully compatible with the PCMCIA release 2.0 standard called the Exchangeable Card Architecture (ExCA). In order to implement card capability in a portable computer, additional BIOS code called Socket Services is minimally required to manage the system card functionality. To implement the specifications, socket services needs an additional 16 Kbyte of code space.

Furthermore, in the pen-based PC applications, there are even greater BIOS requirements to design-in unique features such as: pen extensions, touchscreen capability

and character recognition interface code. These new features require the implementation of additional BIOS code (may be 16 Kbytes to 32 Kbytes).

To take full advantage of Desktop system capabilities and performance while still having a portable computer to take on the road, docking station designs were conceived. This added level of complexity for the portable computer increases the code required in a basic system BIOS.

3.3 Putting Microsoft MS-DOS 5.0 Operating System ROM Version into the BIOS Chip

Additionally, MS-DOS 5, ROM version is now becoming a standard in virtually all diskless sub-notebook, notebook and pen PC implementations. Many factors contribute to this approach. Chief among them is the reduced disk access and the resulting longer battery life. Another factor is the instant boot capability which is essential in certain applications.

Today's MS-DOS 5, ROM version occupies 64 Kbytes of code space as specified from the Microsoft Product Description.

3.4 Relocated Resident VGA Code: VIDEO BIOS

As described above in section 3.0, video BIOS can also be mapped into the system Flash BIOS memory allowing the entire system non-volatile storage requirements to be satisfied with one Flash device. Resident VGA BIOS code takes approximately another 32 Kbytes of memory space.

In summary, adding all the above code size requirements, the resulting BIOS storage area increases from the initial 128 Kbyte requirement to somewhere between 208 Kbytes (without pen extensions) to 240 Kbytes (including a pen input capability).

There are already portable designs today which have filled a 256 Kbyte code space to accommodate some of the above mentioned needs.

4.0 HARDWARE DESIGN FOR A 256 KBYTE BIOS

4.1 Intel 28F200BX/002BX Boot Block Flash Memory Family

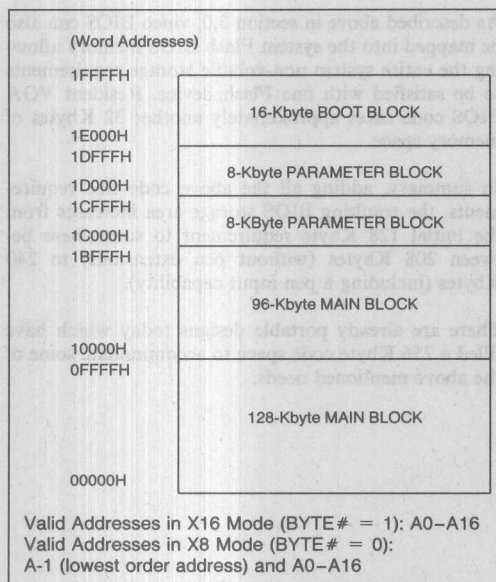
Building upon the wide acceptance of the Intel28F001BX 1 Mbit flash memory for BIOS designs, a new family of higher density flash components is now available to solve the PC designer's need of implementing extended BIOS code beyond 128 Kbytes.

These new flash memories at the 2 Megabit density levels, are structured around the same boot block architecture as the Intel 28F001BX and are therefore compatible. They provide block erasure capability, boot code hardware protection and very low power consumption as in the case of the 28F001BX.

In addition, they incorporate new features to simplify the device interface and allow the system designer to optimize platform designs.

These new features are summarized as follows:

- User selectable 8-bit or 16-bit read/write operation (28F200BX)
- 60ns access time performance
- 16 Kbytes Boot Block space which is hardware protected



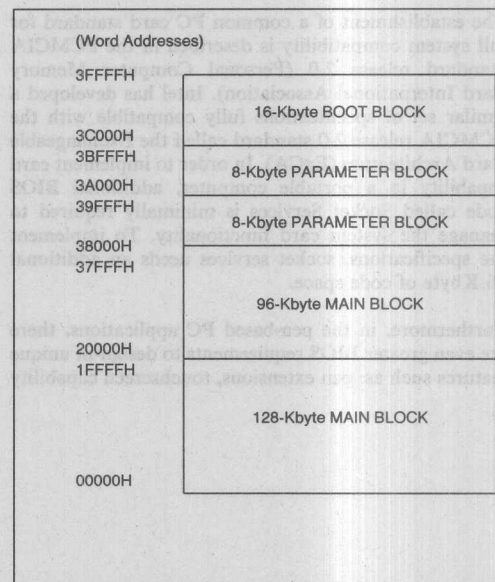
28F200BX-T Top Boot Map

- Two, 8 KBytes Parameter Blocks
- One, 96 KByte Main Block
- One, 128 KByte Main Block
- 8-bit only operation and packaging for space sensitive applications (28F002BX)

In this section, we will describe these new features in more detail and discuss their system applicability.

The blocking scheme, while still of the boot block type, is expanded by defining additional blocks (2 main blocks) to allow for software modularization and a self-contained design.

As the size of the BIOS code stored in any one device grows due to the complexity and high integration of chip-sets, so does the "kernel" code stored in the boot section. The boot and parameter blocks were accordingly doubled in size in comparison to the 28F001BX device. The two parameter blocks of 8 KBytes each allow the PC designer to store BIOS extensions or Battery-Backed SRAM configuration data (CMOS RAM, EISA configuration parameters). The two main blocks are used to store the main BIOS code in modular fashion if so desired for future easy updates. These main blocks can also be used to store ROM-executable Operating System software such as MS-DOS 5, ROM version or drivers and utilities. Refer to Figure 2 for the block locations for both the 28F200BX-T and 28F002BX-T.



28F002BX-T Top Boot Map

Figure 2. 28F200BX-T/002BX-T Memory Maps

In addition, the 28F200BX/002BX devices incorporate new capabilities desired by today's sophisticated PC designers.

The byte-wide or word-wide feature available as a designer-selectable option gives the ability to interface to an 8-bit or 16-bit wide bus. The performance and hardware goals of some systems may require 16-bit BIOS data bus. A system with a small amount of RAM and a large amount of flash memory-based operating system code for example, may require a 16-bit BIOS data bus to maintain a high performance level of operation.

The high access speed of these new devices, which for the first time break the 60 ns barrier at the 2M and 4M densities, is another big advantage the PC designer can fully exploit to increase system performance and acceptance in the marketplace. For example, a PC designer may choose to execute BIOS directly out of flash memory instead of shadowing to system RAM as well as execute MS-DOS 5, ROM version out of flash for instant-on capability and to achieve better overall system performance.

Block erasure allows independent modification of code and data and maximum flexibility in production as well as after the system is shipped. The boot block, which is hardware protected, insures that minimal BIOS code is present to always boot up the system successfully. The 16 Kbyte boot block is protected from alteration during system power excursions by a high voltage pin. This write protection pin called RP# has to transition to 12V with the normal Vpp voltage pin to allow for boot block write and erase operations.

If systems are designed with the ability to jumper or switch RP# to high voltage (12V), guaranteed full non-volatility of the boot code is achieved. This feature always guarantees system recovery from power failure and provides the security needed for the end user when performing BIOS code updates.

To meet the crucial needs of lower power consumption, the 28F200BX/002BX devices incorporate a deep-power down current mode activated through the RP# pin under the TTL/CMOS level control. When this pin transitions to ground the device typically consumes 1 microwatt through the VCC supply pin.

In addition, the 28F200BX/002BX devices include an Automatic Power Savings feature during active mode of operation. This feature allows the memory chip to put itself in a very low current state when it is enabled but not accessing a new memory location.

The 28F200BX/002BX devices incorporate an internal Write State Machine, Command User Interface and a Status Register to fully control the program and erase operations and greatly simplify the user write and erase algorithms and hence the update code procedure. They also include an erase suspend feature which allow the system to service interrupts and access the device during BIOS code updates (refer to Appendix B).

Finally, the 28F200BX/002BX, 2-Megabit devices have an equivalent 4-Megabit boot block flash memory family of devices called the 28F400BX/004BX, allowing for easy density upgrade and total compatibility between systems using both types of memories. Refer to the documentation mentioned in the reference section of this application note.

Figure 3 is a block diagram description of the 28F200BX/002BX products.

4.2 Extended Flash BIOS Design Example

This design example focuses primarily on how to interface the Intel 28F200BX-T or 28F002BX-T Boot Block flash memories to the Intel386SL Microprocessor Superset in a 16-bit wide or an 8-bit wide configuration respectively. **This is an extended BIOS design example which demonstrates how the barrier of the 128-Kbyte BIOS size memory is eliminated. The design principles in this example apply to designs incorporating chip sets interfaced to SL Enhanced CPUs.** Figure 4 shows an interface diagram of the Intel 28F200BX-T Boot Block Flash memory (128 K x 16) to the Intel386SL Microprocessor superset. Figure 5 shows an equivalent interface diagram of the Intel 28F002BX-T Boot Block flash memory (256 K x 8) to the Intel386SL Microprocessor Superset.

The Intel386SL Microprocessor Superset Flash BIOS interface supports up to 256 Kbytes of flash memory BIOS (a 2 Megabit flash memory device) to enable the system designer to meet specific design goals as described in section 3 above.

The Intel386SL Microprocessor Superset supports the following features:

- Up to 256 Kbytes flash memory BIOS
- VGA BIOS mapping into system BIOS
- 8-bit or 16-bit BIOS interface
- Programmable number of flash memory wait states for read access (from zero to fifteen Wait-States to optimize the system performance)
- BIOS shadowing mechanism

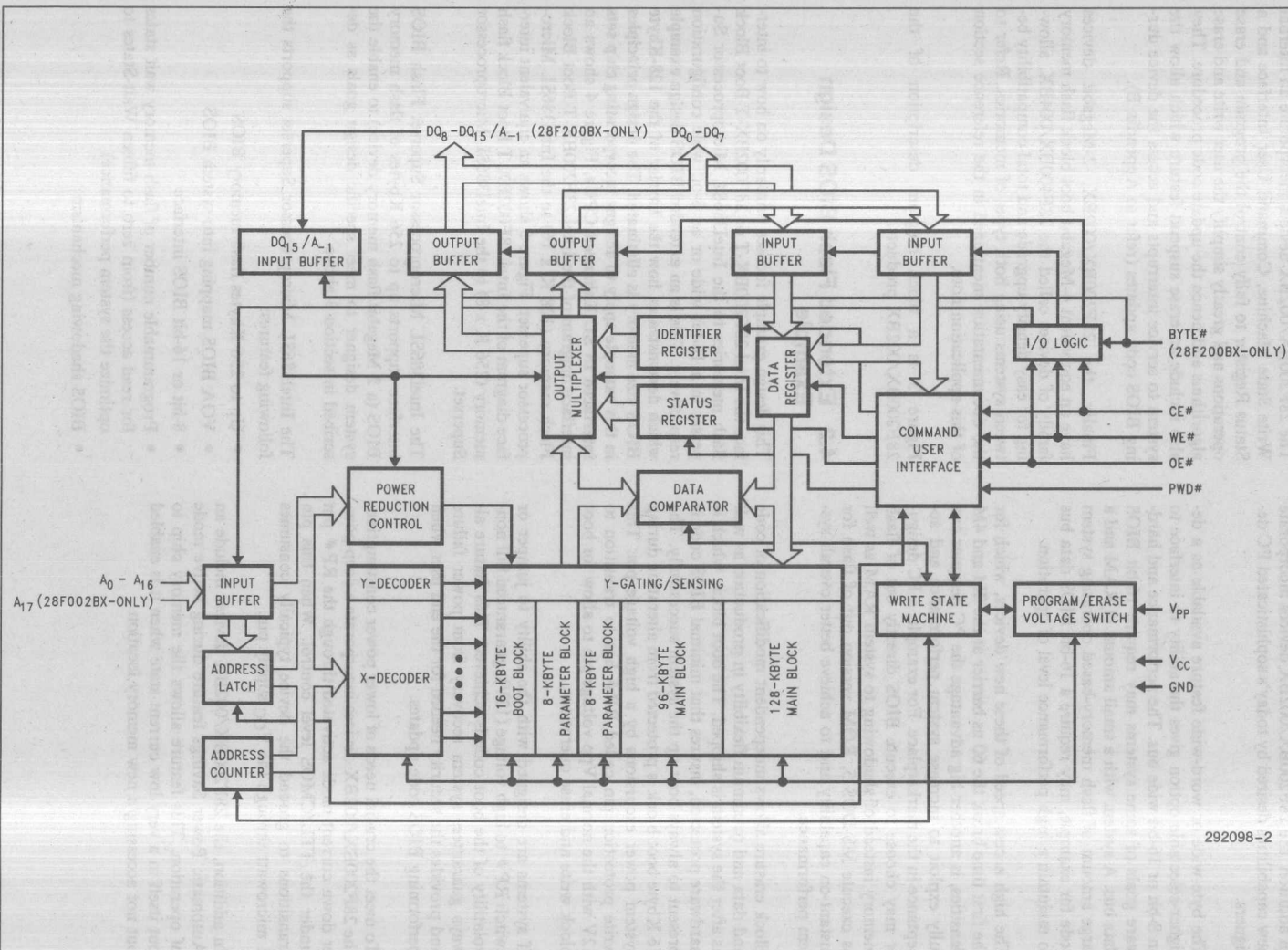


Figure 3. Block Diagram of the 28F200BX/28F002BX

Flash BIOS size configurations in the Intel386SL Microprocessor Superset system are controlled by programming certain registers located in the normal I/O address space.

When a 256 Kbyte Flash BIOS configuration is programmed into the Intel386SL Microprocessor Superset, 128 Kbytes are directly accessible in the E0000H–FFFFFFH address range. The other 128 Kbytes are decoded at the top of the 16th or 32nd Megabyte of the Intel386SL superset address space, i.e., either:

FC0000H–FDFFFFH or 1FC0000H–1FDFFFFH. This extra ROM space is accessed by programming the ISA sliding control register to point to one of these two areas and then accessing the ISA sliding window in the D0000H–DFFFFH address range (64 Kbytes). This mechanism allows complete access of the 256 Kbytes of BIOS code without having to enter the Intel386SL protected mode.

The BIOS code size is used to internally decode the ROM address space and generate two chip select signals: ROMCS0# and ROMCS1#, to control the Flash

BIOS device. In the case of a single Flash BIOS device, only ROMCS0# is needed to drive the CE# chip select signal of the flash memory.

In the diagram of Figure 4 note the following:

- BYTE# signal is set high to enable 16-bit operation of the flash device.
- The highest order system address line SA₁₆ is inverted when FLIP# signal becomes active at boot-up time to relocate the boot kernel code at the top of the 1 Mbyte memory map for the system to boot from it. (See section 4.5).
- ROM16/8# is set high to enable 16-bit bus operations.
- RP# signal is gated by the PWRGOOD signal (for reset when power fails) and by system RESET signal.
- V_{PP} supply voltage is switched to the flash device only when BIOS updates are required.

In addition to the above considerations, note the following in Figure 5:

- The highest order system address line SA₁₇ is also inverted when FLIP# signal becomes active at system boot-up time.
- ROM16/8# is set low to enable 8-bit only bus operations.

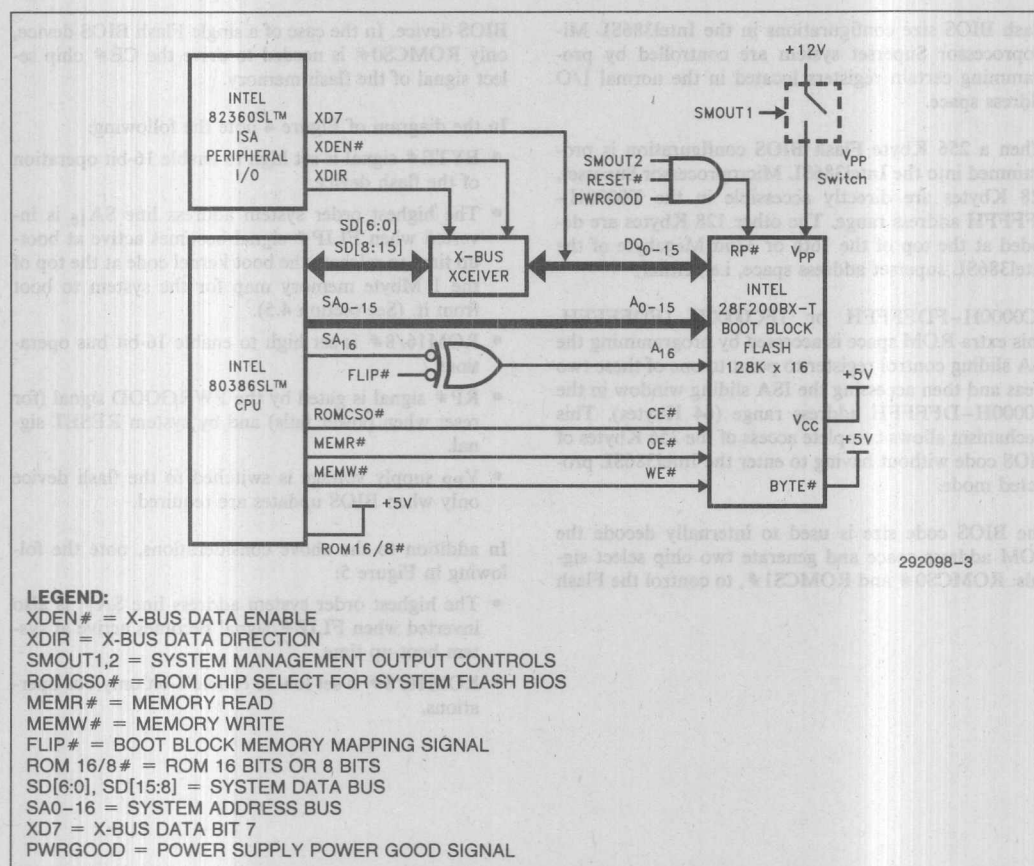


Figure 4. Key Attributes of this Optimized BIOS Interface should be Incorporated in BIOS Designs > 128 Kbytes.

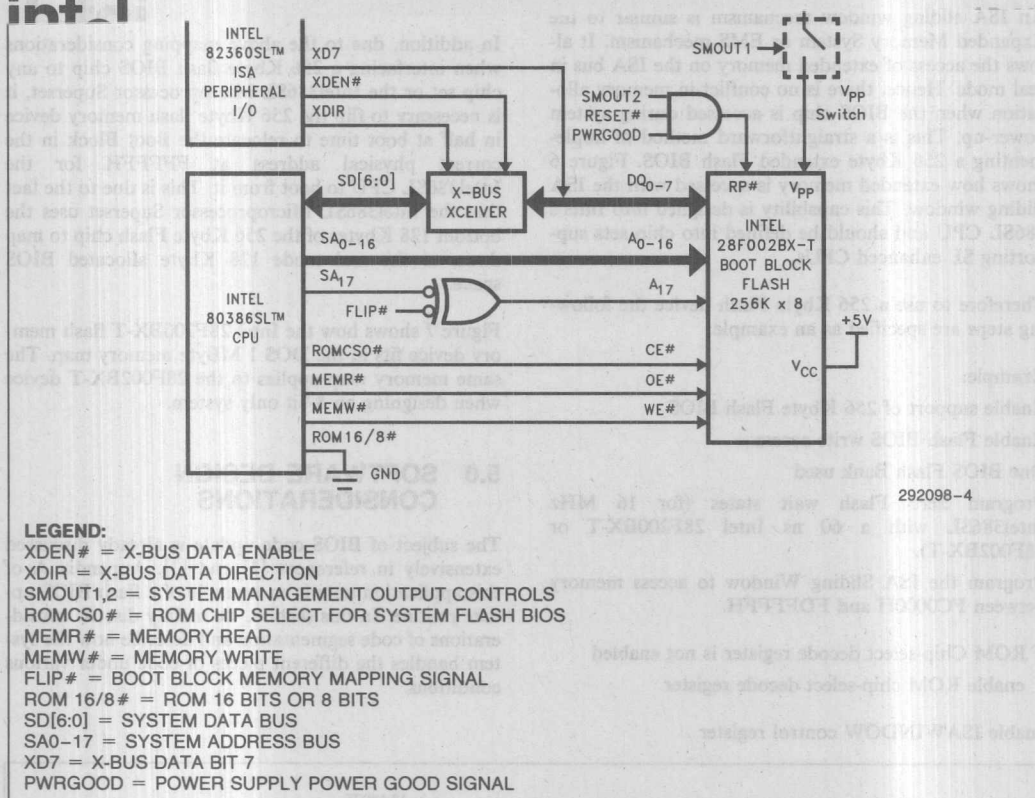


Figure 5. Key Attributes of this Optimized BIOS Interface should be Incorporated in BIOS Designs > 128 Kbytes.

4.3 ISA Sliding Window

An ISA sliding window mechanism is similar to the Expanded Memory System or EMS mechanism. It allows the access of extended memory on the ISA bus in real mode. Hence, there is no conflict in memory allocation when the BIOS chip is accessed during system power-up. This is a straightforward method of implementing a 256 Kbyte extended Flash BIOS. Figure 6 shows how extended memory is accessed with the ISA sliding window. This capability is designed into Intel's 386SL CPU and should be defined into chip sets supporting SL enhanced CPUs.

Therefore to use a 256 Kbyte Flash device the following steps are specified as an example:

Example:

Enable support of 256 Kbyte Flash BIOS

Enable Flash BIOS write access

One BIOS Flash Bank used

Program zero Flash wait states (for 16 MHz Intel386SL with a 60 ns Intel 28F200BX-T or 28F002BX-T).

Program the ISA Sliding Window to access memory between FC0000H and FDFFFFH.

if ROM Chip-select decode register is not enabled

enable ROM chip-select decode register

enable ISAWINDOW control register

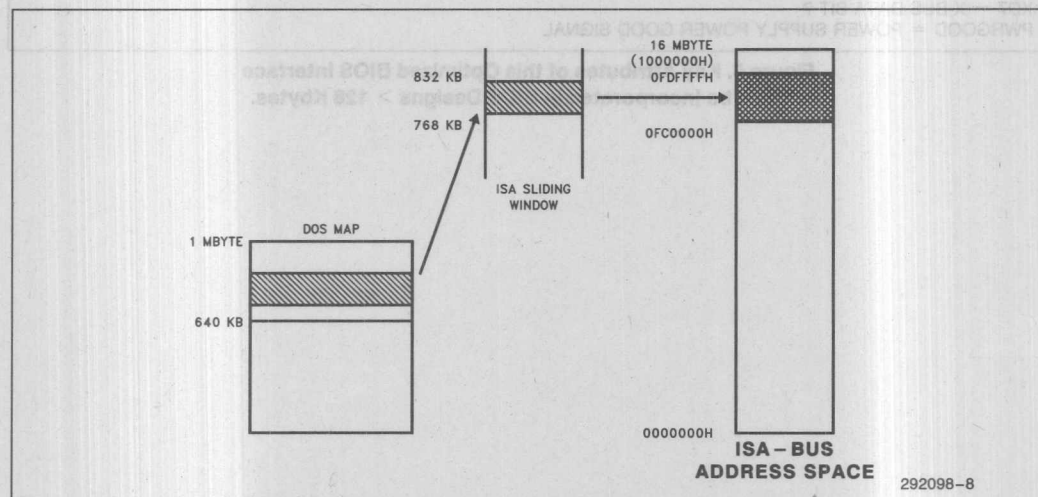


Figure 6. ISA Sliding Window and Extended Memory Maps

4.4 The 28F200BX-T/28F002BX-T in the 1 Mbyte DOS Memory Map

In addition, due to the above mapping considerations when interfacing a 256 Kbyte flash BIOS chip to any chip set or the Intel386SL Microprocessor Superset, it is necessary to flip the 256 Kbyte flash memory device in half at boot time to relocate the Boot Block in the correct physical address at FFFFFH for the Intel386SL CPU to boot from it. This is due to the fact that the Intel386SL Microprocessor Superset uses the bottom 128 Kbytes of the 256 Kbyte Flash chip to map down to the real mode 128 Kbyte allocated BIOS space.

Figure 7 shows how the Intel 28F200BX-T flash memory device fits in the DOS 1 Mbyte memory map. The same memory map applies to the 28F002BX-T device when designing an 8-bit only system.

5.0 SOFTWARE DESIGN CONSIDERATIONS

The subject of BIOS code update is already discussed extensively in references [1] and [3]. Appendix A of this application note is an example of a flash BIOS update routine. In this section, we mainly discuss considerations of code segmentation and describe how the system handles the different pieces of code under various conditions.

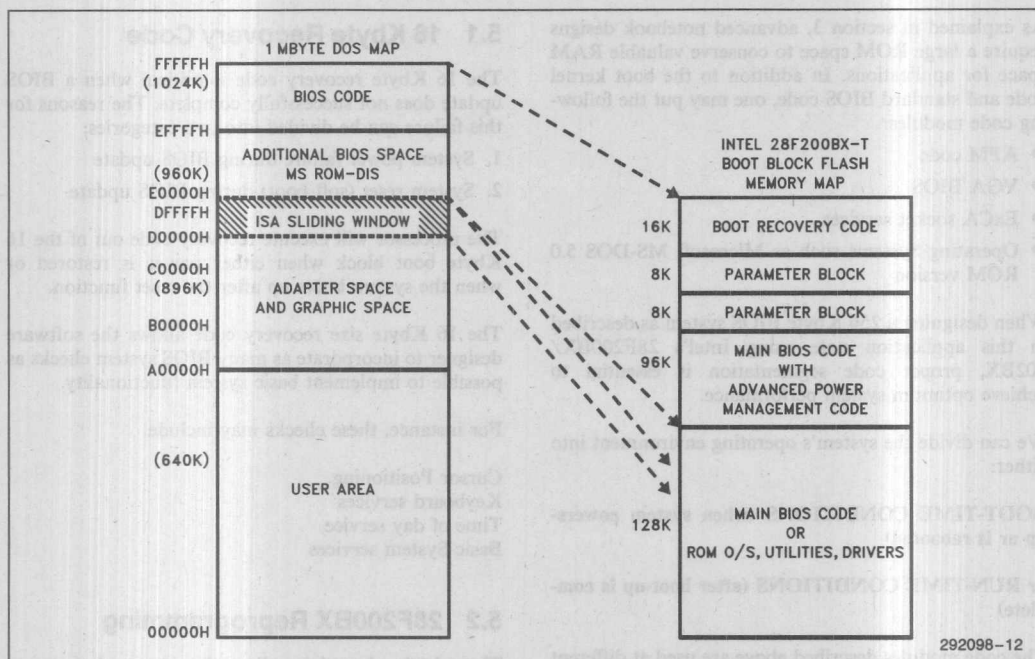


Figure 7. 28F200BX-T/28F002BX-T in the 1 Mbyte DOS Memory Map

As explained in section 3, advanced notebook designs require a large ROM space to conserve valuable RAM space for applications. In addition to the boot kernel code and standard BIOS code, one may put the following code modules:

- APM code
- VGA BIOS
- ExCA socket services
- Operating Systems such as Microsoft MS-DOS 5.0 ROM version

When designing a 256 Kbyte BIOS system as described in this application note using Intel's 28F200BX/002BX, proper code segmentation is essential to achieve optimum system performance.

We can divide the system's operating environment into either:

BOOT-TIME CONDITIONS (when system powers-up or is rebooted)

or **RUN-TIME CONDITIONS** (after boot-up is complete)

The code modules described above are used at different times during normal system operation. Hence, a segregation of the code stored in the flash memory is necessary for proper system operation.

BOOT-TIME EXECUTION

The system requires the boot kernel to be present at the FFFFFH segment during this portion of the cycle.

When using the 28F200BX/002BX flash memory, the first page present will be the top 128 Kbyte. The rest of the first page is used to store APM code, VGA BIOS code or ExCA socket services. These code modules are then copied from their location below the boot block to system RAM for later initialization after the standard BIOS has started.

Once all code modules are copied into RAM, this first 128 Kbyte page of flash can be swapped out or exchanged with the 2nd 128 Kbyte page which is required for run-time execution.

RUN-TIME EXECUTION

Standard BIOS code is required to be present from E0000H to FFFFFH segment during this time period to handle any BIOS calls and maintain proper system operation.

Microsoft MS-DOS 5.0 ROM version code is also required to be present so the BIOS can "SCAN" it in as an adapter.

5.1 16 Kbyte Recovery Code

The 16 Kbyte recovery code is critical when a BIOS update does not successfully complete. The reasons for this failure can be divided into two categories:

1. System power failure during BIOS update
2. System reset (soft boot) during BIOS update

The processor will execute recovery code out of the 16 Kbyte boot block when either power is restored or when the system boots up after the reset function.

The 16 Kbyte size recovery code allows the software designer to incorporate as many BIOS system checks as possible to implement basic system functionality.

For instance, these checks may include:

Cursor Positioning
Keyboard services
Time of day service
Basic System services

5.2 28F200BX Reprogramming

Three basic algorithms allow the system designer to reprogram the Flash BIOS chip and perform the necessary tasks for a BIOS update. These algorithms are:

Automated Byte/Word-wide programming
Automated Block erase
Erase Suspend and Resume

For a description of these algorithms, the reader is encouraged to study reference [5]. Appendix B in this application note includes the four flowcharts associated with the algorithms.

To obtain the software drivers necessary to control the device reprogramming operations, consult your local Intel sales office.

5.3 Power Management

Power management is an essential part of any true portable PC design. The design of sophisticated power management techniques is becoming a key differentiator between different machines, and hence is a competitive advantage for the system integrator.

To help the system designer with this often difficult task of optimizing system performance and battery life, the 28F200BX family of products includes three distinct low power modes of operation. These are:

- Standby Current Mode, where the device typically consumes 50 μ A

- Automatic Power Savings feature, where the device typically consumes 1 mA
- Deep powerdown Mode, where the device typically consumes 0.2 μ A

For a detailed description of these modes of operation, consult reference [8].

6.0 DESIGNING A 3.3V SYSTEM

The ability to design 3.3V systems used to be a future consideration. Longer battery life and lighter weight portable computers are some of the key objectives for any portable design. Now, thanks to the increasing availability of low voltage components, true low power machines are possible to realize in practice.

6.1 Low Voltage Chips

The list of 3.3V components available to build the essential parts of a portable computer is becoming longer every day. Semiconductor manufacturers have recognized the urgent need to supply low voltage chips to the portable marketplace.

The Intel 28F200BX/002BX Boot Block Flash Memories are available in 3.3V versions. These low voltage versions of the 28F200BX/002BX are functionally equivalent to their 5V counterparts and are 100% pin-out compatible. So a system converting to 3.3V operation can substitute low voltage 2 Mbit chips (28F200BX-L/002BX-L) when desired without any circuit board modification.

6.2 Power Savings and Improved Battery Life

The 28F200BX/002BX 3.3V chips reduce the total power drawn during normal read operation to less than 25% of the total power in 5V mode. This is a substantial savings in current which translates to 25% less battery drain and hence longer battery life.

Similarly, low voltage version of the most popular microprocessors reduce the total power dissipated by a substantial amount.

The combination of these current savings plus the other system components current reductions improve battery life dramatically and allow the mobile PC user to benefit from weight reduction, longer operating time, lower system cost and higher performance.

7.0 CONCLUSIONS

7.1 Benefits of Extended Flash BIOS

This application note deals with the concepts of extended BIOS implementations in portable PC designs, but it can also be easily adapted to desktop PC systems for which BIOS code requirements can easily exceed 128 Kbytes.

We have attempted to explain the requirements and the needs of today's advanced portable BIOS designs which have to meet many difficult and often conflicting requirements.

Boot Block flash memory is the ideal storage solution to implement the above mentioned features. Furthermore, as the cost of solid state non-volatile flash memory keeps decreasing, the need to switch to these types of media for storing extended BIOS, operating system software, utilities, and in the future application code, becomes more evident and perhaps the only way a PC manufacturer can effectively compete by producing the best engineered, most optimally designed notebooks, palmtop PCs and pen-based computers.

REFERENCES

For more information on the concepts presented in this application note, the reader is encouraged to reference the following documents.

- [1] Technical Paper: "Flash: The Optimum BIOS Storage Device" by Brian Dipert, 1991 SVPC — Order Number 297003
- [2] "ROM BIOS: The best place for portable PC Power-Management features" by Lance Hansche, 1991 SVPC
- [3] AP-341: "Designing an Updatable BIOS Using Flash Memory"—Order Number 292077
- [4] AP-357: "Power Supply Solutions for Flash Memory,"—Order Number 292092
- [5] Intel 28F200BX/002BX Datasheet—Order Number 290448
- [6] Intel 28F400BX/004BX Datasheet—Order Number 290451
- [7] Intel 28F200BX-L/002BX-L Datasheet—290449
- [8] Intel 28F400BX-L/004BX-L Datasheet—290450

APPENDIX A

Example of a Flash Update Utility Pseudo-Code

This example is for a standard BIOS code and APM code update using the 28F200BX/002BX flash device. Modify this utility, if required, to suit your particular system needs.

Initialize system (set-up user screen, check battery power, check device ID)

Get BIOS/APM file Options (from floppy or through modem)

If no file present
Send error message to insert BIOS update floppy, or press ESC to exit

Display BIOS/APM update files, prompt user for choice and load to memory

If file is invalid
Prompt for correct file or exit

Inform user of upcoming event, provide option to continue or exit

If user continues, inform user not to turn off the power or soft-reboot system (CNTL-ALT-DEL)

Erase 28F200BX/002BX device 96-Kbyte main block

If system interrupt occurs
Suspend erase operation if flash memory access is required

Resume erase operation of 96-Kbyte main block

Write new file(s) into flash memory 96-Kbyte main block

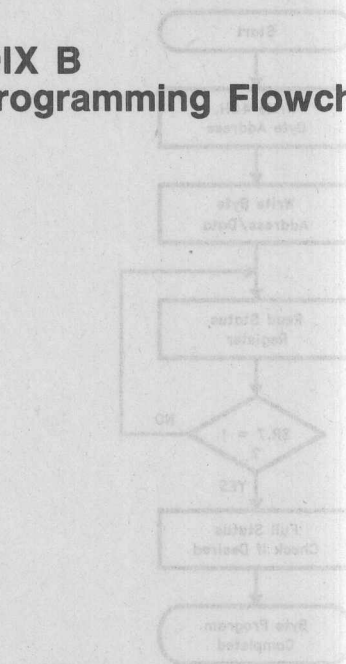
Indicate to user that flash reprogramming is complete

Prompt user to reboot the system to continue normal operation

APPENDIX B **28F200BX-T/28F002BX-T Programming Flowcharts**

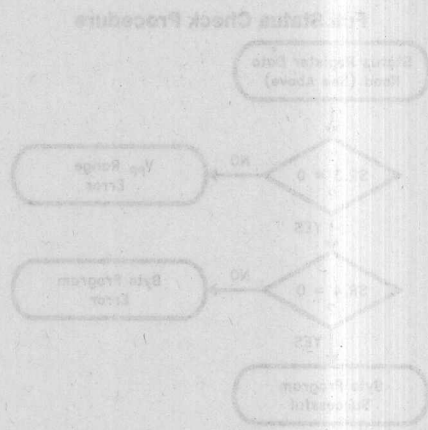
Comments	Command	Bus Direction
Program Address = bits to be programmed	Program	Write
Data to be programmed Address = bits to be programmed	Program	Write
Blank register data Upper OE is 12K to update Status Register	Read	Read
Check SR 1 1 = Ready0 = Busy	Standby	Standby

Repeat for subsequent data.
Full status check can be done after each byte of data sequence of bytes.
Write FRH after the last byte programming instruction to return the device to Read Array mode.

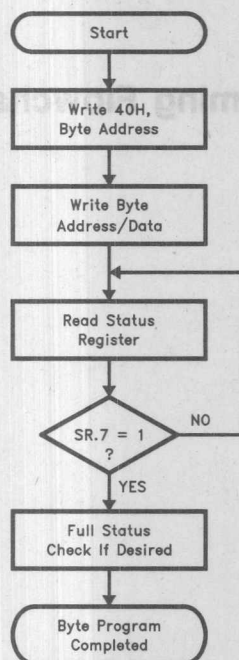


Comments	Command	Bus Direction
Check SR 2 1 = Vpp On Delay	Standby	Standby
Check SR 4 1 = Byte Program Error	Standby	Standby

SR 2 MUST be cleared, it set during program attempt, before further attempts are allowed by the Write Status Machine.
SR 4 is only cleared by the Check Status Register Command in cases where multiple bytes are programmed before full status is checked.
If error is detected, clear the Status Register before attempting into or other error recovery.



Automated Byte Programming Flowchart



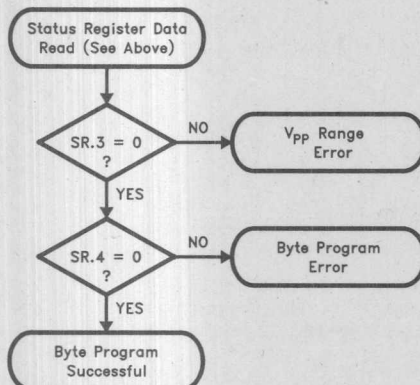
292098-13

Bus Operation	Command	Comments
Write	Setup Program	Data = 40H Address = Byte to be programmed
Write	Program	Data to be programmed Address = Byte to be programmed
Read		Status Register Data. Toggle OE# or CE# to update Status Register
Standby		Check SR.7 1 = Ready, 0 = Busy

Repeat for subsequent bytes.

Full status check can be done after each byte or after a sequence of bytes.

Write FFH after the last byte programming operation to reset the device to Read Array Mode.

Full Status Check Procedure

292098-14

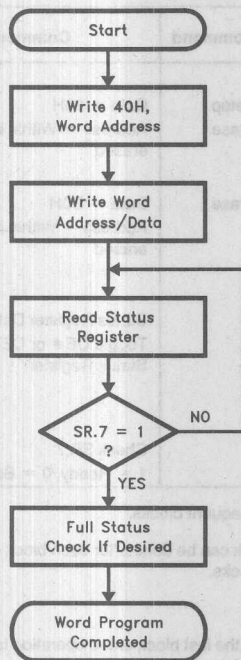
Bus Operation	Command	Comments
Standby		Check SR.3 1 = Vpp Low Detect
Standby		Check SR.4 1 = Byte Program Error

SR.3 MUST be cleared, if set during a program attempt, before further attempts are allowed by the Write State Machine.

SR.4 is only cleared by the Clear Status Register Command, in cases where multiple bytes are programmed before full status is checked.

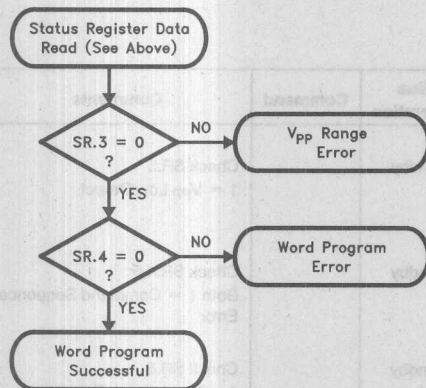
If error is detected, clear the Status Register before attempting retry or other error recovery.

Automated Byte Programming Flowchart



292098-15

Full Status Check Procedure



292098-16

Bus Operation	Command	Comments
Write	Setup Program	Data = 40H Address = Word to be programmed
Write	Program	Data to be programmed Address = Word to be programmed
Read		Status Register Data. Toggle OE# or CE# to update Status Register
Standby		Check SR.7 1 = Ready, 0 = Busy

Repeat for subsequent words.

Full status check can be done after each word or after a sequence of words.

Write FFH after the last word programming operation to reset the device to Read Array Mode.

4

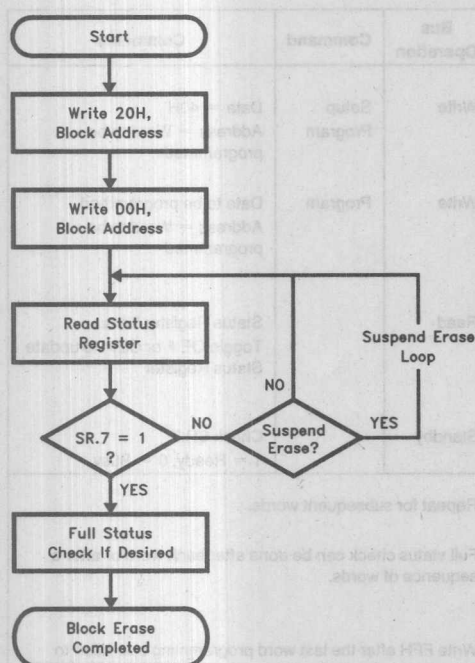
Bus Operation	Command	Comments
Standby		Check SR.3 1 = V _{pp} Low Detect
Standby		Check SR.4 1 = Word Program Error

SR.3 MUST be cleared, if set during a program attempt, before further attempts are allowed by the Write State Machine.

SR.4 is only cleared by the Clear Status Register Command, in cases where multiple words are programmed before full status is checked.

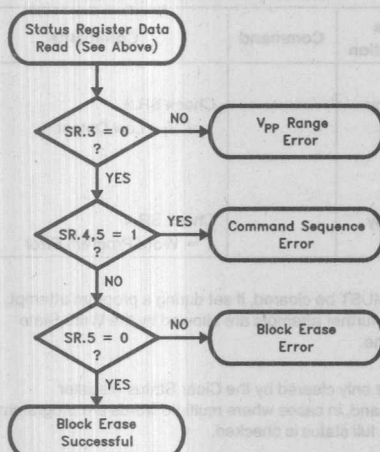
If error is detected, clear the Status Register before attempting retry or other error recovery.

Automated Word Programming Flowchart



292098-17

Full Status Check Procedure



292098-18

Bus Operation	Command	Comments
Write	Setup Erase	Data = 20H Address = Within block to be erased
Write	Erase	Data = D0H Address = Within block to be erased
Read		Status Register Data. Toggle OE# or CE# to update Status Register
Standby		Check SR.7 1 = Ready, 0 = Busy

Repeat for subsequent blocks.

Full status check can be done after each block or after a sequence of blocks.

Write FFH after the last block erase operation to reset the device to Read Array Mode.

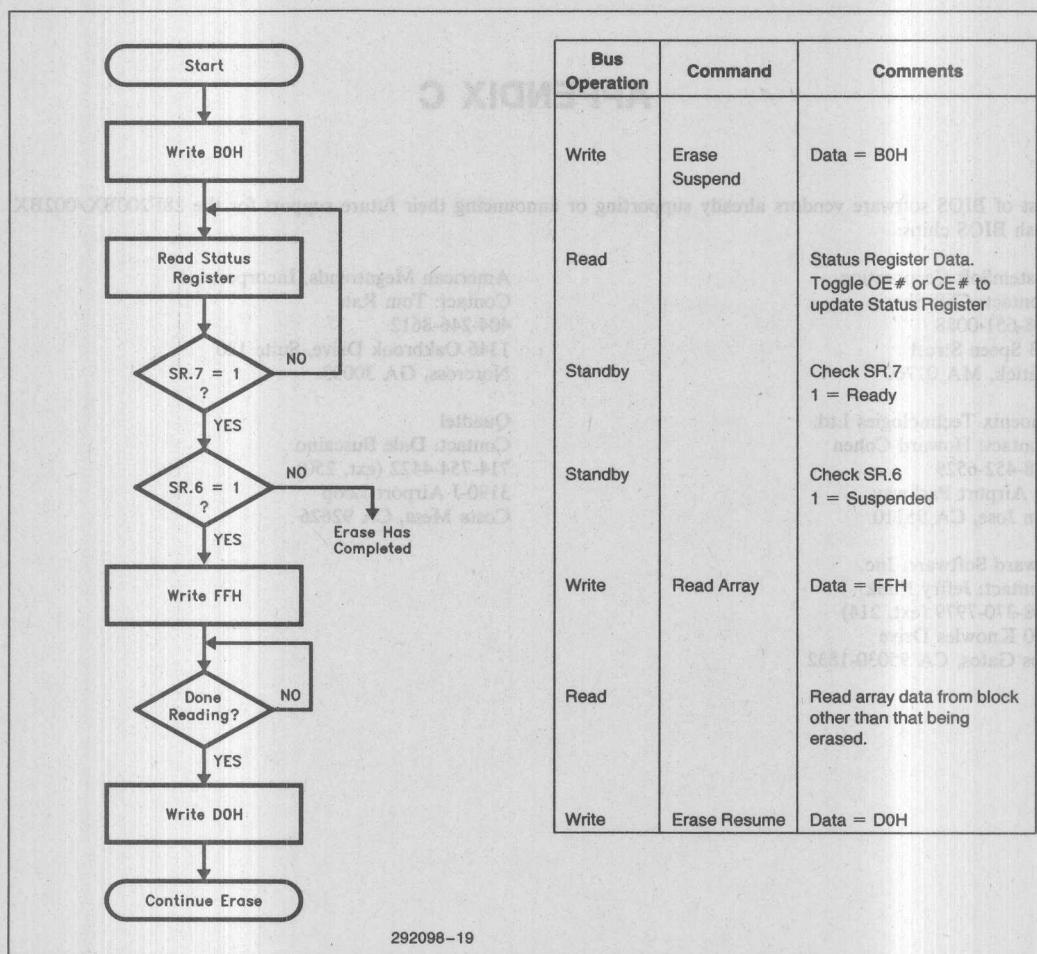
Bus Operation	Command	Comments
Standby		Check SR.3 1 = Vpp Low Detect
Standby		Check SR.4,5 Both 1 = Command Sequence Error
Standby		Check SR.5 1 = Block Erase Error

SR.3 MUST be cleared, if set during an erase attempt, before further attempts are allowed by the Write State Machine.

SR.5 is only cleared by the Clear Status Register Command, in cases where multiple blocks are erased before full status is checked.

If error is detected, clear the Status Register before attempting retry or other error recovery.

Automated Block Erase Flowchart



Erase Suspend/Resume Flowchart

APPENDIX C

List of BIOS software vendors already supporting or announcing their future support for the 28F200BX/002BX flash BIOS chips:

SystemSoft Corporation
Contact: Cliff Sharin
508-651-0088
313 Speen Street
Natick, MA 01760

Phoenix Technologies Ltd.
Contact: Howard Cohen
408-452-6529
40 Airport Parkway
San Jose, CA 95110

Award Software, Inc.
Contact: Jeffry Flink
408-370-7979 (ext. 214)
130 Knowles Drive
Los Gatos, CA 95030-1832

American Megatrends, Incorporated
Contact: Tom Rau
404-246-8612
1346 Oakbrook Drive, Suite 120
Norcross, GA 30093

Quadtel
Contact: Dale Buscaino
714-754-4422 (ext. 250)
3190-J Airport Loop
Costa Mesa, CA 92626

ENGINEERING REPORT

The Intel 28F001BX-T and 28F001BX-B Flash Memories

4

BRIAN DIPERT
OWEN JUNGROTH
MEMORY COMPONENTS DIVISION

October 1993

Order Number: 294010-003

4-273

The Intel 28F001BX-T and 28F001BX-B Flash Memories

CONTENTS

	PAGE
INTRODUCTION	4-275
TECHNOLOGY OVERVIEW	4-275
DEVICE ARCHITECTURE	4-276
Write State Machine and Command/Status Registers	4-276
Internal Oscillator	4-277
Supply Voltage Sensing	4-277
Erase	4-278

CONTENTS

	PAGE
Programming	4-279
Reset-Power Down	4-279
DEVICE RELIABILITY	4-280
Cell Margining	4-280
Erase/Program Cycling	4-281
SUMMARY	4-281

INTRODUCTION

Intel's 28F001BX ETOX (EPROM tunnel oxide) flash memories add selective block erasure, an integrated Write State Machine and powerdown capability to Intel's standard flash memory product line. Flash memory enhances EPROM non-volatility and ease of use through electrical erasure and reprogramming. Advances in tunnel oxides and photolithography have made it possible to develop a double-polysilicon single-transistor read/write random access nonvolatile memory, capable of greater than 100,000 reprogramming cycles. The 28F001BX flash memories electrically erase all bits in a block matrix via electron tunneling. The EPROM programming mechanism of hot electron injection is employed for electrical byte programming.

A Command Register/Status Register interface to a Write State Machine, internal margin voltage generation, power up/down protection and address/data latches augment standard EPROM circuitry to optimize Intel's 28F001BX family for microprocessor-controlled reprogramming.

Read timing parameters are equivalent to those of CMOS EPROMs, EEPROMs and SRAMs. The 120 ns access time results from a memory cell current of approximately 50 μ A, low resistance poly-silicide wordlines, advanced scaled periphery transistors and an optimized data-out buffer.

The dense one-transistor cell structure, coupled with high array efficiency, yields a one megabit die measuring 235 by 268 mils.

TECHNOLOGY OVERVIEW

Intel's ETOX flash memory technology is derived from its standard CMOS EPROM process base. Using

advanced 1.0 μ m double-polysilicon n-well CMOS technology, the 131,072 x 8 bit flash memories employ a 3.8 μ m x 4.0 μ m single transistor cell, affording equivalent array density as comparable EPROM technology. The flash memory cell structure is identical to the EPROM structure, except for the thinner gate (tunnel) oxide. Figure 1 compares the flash memory cell to the EPROM cell.

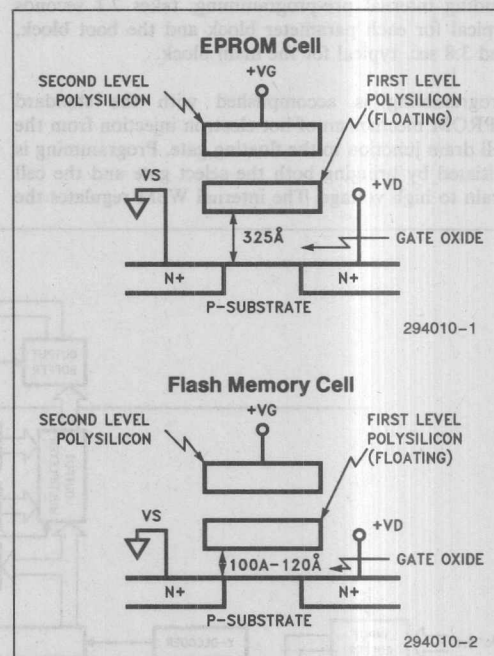


Figure 1. EPROM Cell vs. Flash Memory Cell

High quality tunnel oxide under the single floating polysilicon gate promotes electrical erasure. All cells of a given block are simultaneously erased via Fowler-Nordheim tunneling. Applying 12V on the block source junctions and grounding the select gates erases a given block. The internal Write State Machine (WSM) controls the erase algorithm, including block pre-programming before erasure. WSM-controlled erasure, including internal pre-programming, takes 2.1 seconds typical for each parameter block and the boot block, and 3.8 sec. typical for the main block.

Programming is accomplished with the standard EPROM mechanism of hot electron injection from the cell drain junction to the floating gate. Programming is initiated by bringing both the select gate and the cell drain to high voltage. The internal WSM regulates the

internal program algorithm after the correct command sequence is written to the 28F001BX. Typical program time is 18 μ s per byte.

DEVICE ARCHITECTURE

Write State Machine and Command/Status Registers

Intel's 28F001BX flash memories contain an on-chip Write State Machine that automatically controls erase and program algorithms, dramatically simplifying user interface. Figure 2 shows the 28F001BX block diagram.

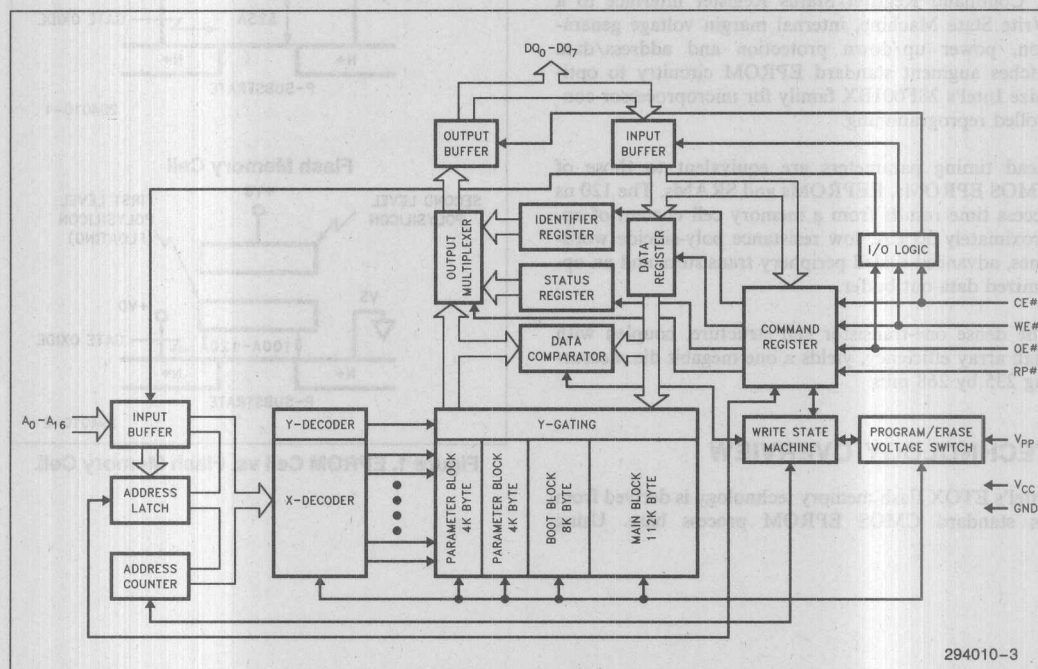


Figure 2. 28F001BX Block Diagram

erase, program, status register read/clear, ID read and array read operations, without the need for additional control pins or the multiplexing of high voltage with control functions. The WSM, with its integrated oscillator, performs a majority of the standard flash memory program and erase algorithms automatically. This makes system timers no longer necessary and frees the system to service interrupts or perform other functions during device erase or program. On-chip address and data latches minimize system interface logic and free the system bus. The Write State Machine accepts array read, ID read and Status Register read and clear commands whenever power is applied to the 28F001BX. High voltage (12V) on V_{PP} additionally enables successful program and erase.

The WSM consists of a Command Register, Status Register, State Machine, oscillator, command decoder, data latch and address latch. The command decoder output feeds the State Machine, enabling the high voltage flash-erase switch, program voltage generator and erase/program verify voltage generator.

Functions are selected via the Command Register in a microprocessor write cycle controlled by the Chip Enable ($CE\#$) and Write Enable ($WE\#$) pins. The rising edge of $WE\#$ latches the address and data-in registers, and initiates an operation. Status Register contents are driven to the outputs on the falling edge of $CE\#$ or Output Enable ($OE\#$), whichever occurs last in the read cycle.

Internal Oscillator

The Write State Machine is designed using clocked logic circuits. An on chip ring oscillator generates the clock signals. The frequency of a standard ring oscillator varies with processing, temperature and supply voltage. The improved design used on the 28F001BX minimizes these variations.

The switching current of each stage in the ring oscillator is set by a current reference. This reference current varies linearly with V_{CC} . The trip point of each ring oscillator inverter also varies linearly with V_{CC} . These two effects essentially cancel each other out and the resulting oscillator period is proportional to RC , with only a small dependence on V_{CC} .

The value of R is set by an on chip resistor. The value of C is set by the gate capacitance of the inverters in the ring oscillator. Process variations in the values are reduced by trimming the period of each oscillator during manufacturing. The resistor is the only source of temperature variation.

temperature and supply voltage. The circuit works for supply voltages outside the normal operating conditions and for military temperatures.

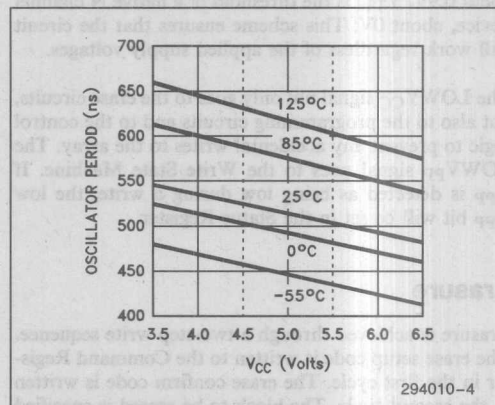


Figure 3. Internal Oscillator Frequency vs Supply Voltage and Temperature

Supply Voltage Sensing

The circuit that generates $LOWV_{CC}$ and $LOWV_{PP}$ is shown in Figure 4. Power supply voltages V_{CC} and V_{PP} are divided down and compared to a reference voltage. If the reference voltage is greater than the divided power supply voltage, the $LOWV_{CC}$ or $LOWV_{PP}$ signal will be pulled high. The V_{REF} level generated by the voltage reference is independent of the supply voltage to the first order.

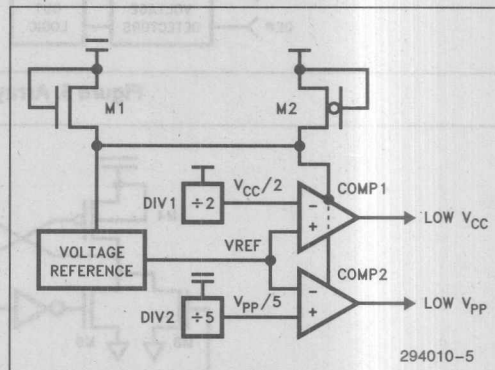


Figure 4. Low Power Detector Circuit

The positive power supply to the circuit is provided by M1 and M2. The source of M1 and M2 will be pulled up to the maximum of ($V_{PP}-V_{TN}$) and ($V_{CC}-V_{TW}$). V_{TN} is the threshold of an implanted N channel device, about 0.9V. V_{TW} is the threshold of a native N channel device, about 0V. This scheme ensures that the circuit will work regardless of the applied supply voltages.

The $LOWV_{CC}$ signal not only goes to the erase circuits, but also to the programming circuits and to the control logic to prevent any accidental writes to the array. The $LOWV_{PP}$ signal goes to the Write State Machine. If V_{PP} is detected as being low during a write, the low V_{PP} bit will be set in the Status Register.

Erase

Erase is achieved through a two-step write sequence. The erase setup code is written to the Command Register in the first cycle. The erase confirm code is written in the second cycle. The block to be erased is specified by writing both commands to any address within the

block. The address is latched and decoded internally by the 28F001BX, and erase of the desired block is subsequently enabled. The rising edge of this second $WE\#$ pulse initiates the erase operation. The boot block will not erase unless the $RP\#$ or $OE\#$ signal is brought to high voltage V_{HH} .

The State Machine triggers the high voltage flash-erase switch, connecting the 12V supply to the source of all bits in the specified block, while all wordlines are grounded. The organization of the block source switches is shown in Figure 5. Fowler-Nordheim tunneling results in the simultaneous erasure of all bits in the addressed block.

The block source switch controls the source voltage of the bits in a particular block. This circuit is shown in Figure 6. During erase, M2 is off and M1 pulls the source to V_{PP} . When not in erase, M1 is off and M2 pulls the source to ground. The high voltage latch formed by M4-M7 converts the low voltage ERASE signal to a high voltage signal that turns M1 off or on.

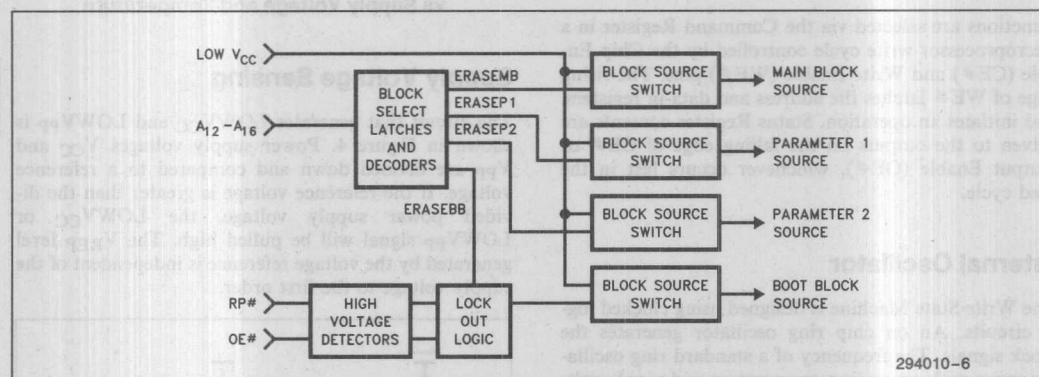


Figure 5. Array Erase Blocking

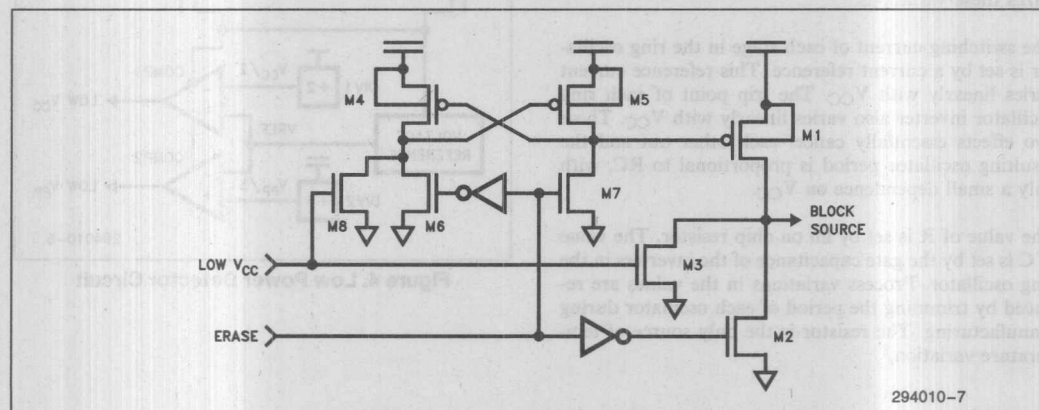


Figure 6. Block Source Switch

The tunneling that occurs during erase requires only a small amount of current. However, the grounded gate initial erase current that occurs on the source of every bit in the array is large. M1 is made large enough to supply this current and still keep the voltage on the source high enough for fast erase time.

The LOWV_{CC} signal protects the array from being erased when V_{pp} is at a high voltage but V_{CC} is a low voltage. When this occurs, M3 will pull the block source to ground. The high voltage latch will be forced into the state that turns M1 off by M8.

After receiving the erase command sequence, the WSM automatically controls block precondition (programming of all bytes to 00H within the chosen block), erase pulses and pulse repetition, timeout delays and byte-by-byte verification of all block addresses using the internally-generated erase margin voltage. The internal erase and verify operations continue until the entire block is erased. System software need only poll the Status Register to determine when the WSM has successfully completed the erase algorithm.

Programming

Programming follows a similar flow. The program set-up command is written to the Command Register on the first cycle. The second cycle loads the address and data latches. The rising edge of the second WE# pulse initiates programming by applying high voltage to the gates and drains of the bits to be programmed.

As with erasure, the WSM controls program pulses and pulse repetition, timeout delays and byte verification. Program and program verify (at the internally-generated verify voltage) continue until the byte is programmed. System software, polling the Status Register, is informed of programming state thru specific status bits.

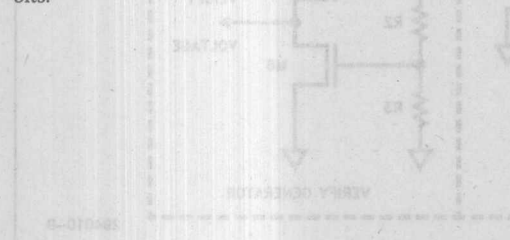


Figure 8. Erase/Program Verify Generator

Reset-Power Down

The 28F001BX has a deep power down mode that reduces I_{CC} and I_{pp} to typically 0.05 μ A and 0.8 μ A, respectively. When RP# is low, the part is in deep power down mode. When RP# is high, the part can be placed in an active or standby mode by state of the CE# pin.

The deep power down mode is similar to the standby mode except that more circuits are turned off. This means that much less power is consumed; it also means that it takes longer for the part to transition into the active mode.

A diagram of the power down circuit is shown in Figure 7. The TTL buffer formed by M1-M3 enables the low power detect circuits, the redundancy address flash bits and the CE# TTL buffer formed by M4-M6. In previous Intel flash chips these circuits were always enabled. The time for these circuits to turn on determines the RP# access time and write specifications.

RP# will function properly with TTL level inputs. However, to get the lowest possible power consumption, full CMOS levels should be used. If voltage on the gate of M3 raises above its threshold voltage of 0.9V, it will turn on and draw current. Input voltages in the 0.7-0.9 range could cause enough subthreshold conduction in M3 to exceed the deep power down current specification. This is why the input voltage for RP# is specified as GND \pm 0.2V.

The use of RP# during system reset is important with automated write/erase devices. When the system comes out of reset it expects to read from the flash memory. Automated flash memories provide status information when accessed during write/erase modes. If a CPU reset occurs with no flash memory reset, proper CPU initialization would not occur because the flash memory would be providing the status information instead of array data. Intel's Flash Memories allow proper CPU initialization following a system reset through the use of the RP# input. In this application RP# is controlled by the same RESET# signal that resets the system CPU.

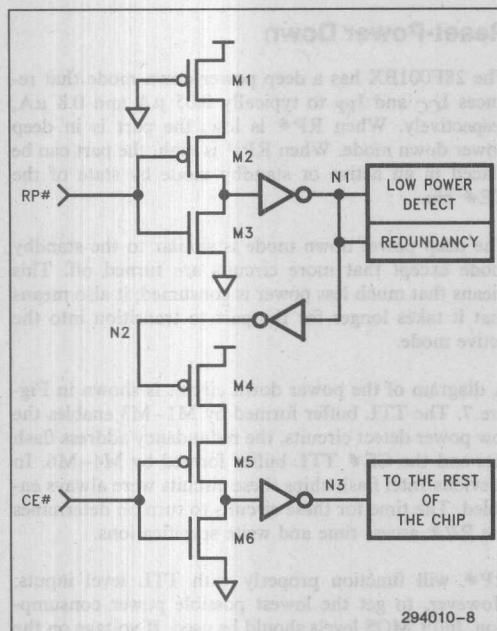


Figure 7. Power Down Circuits

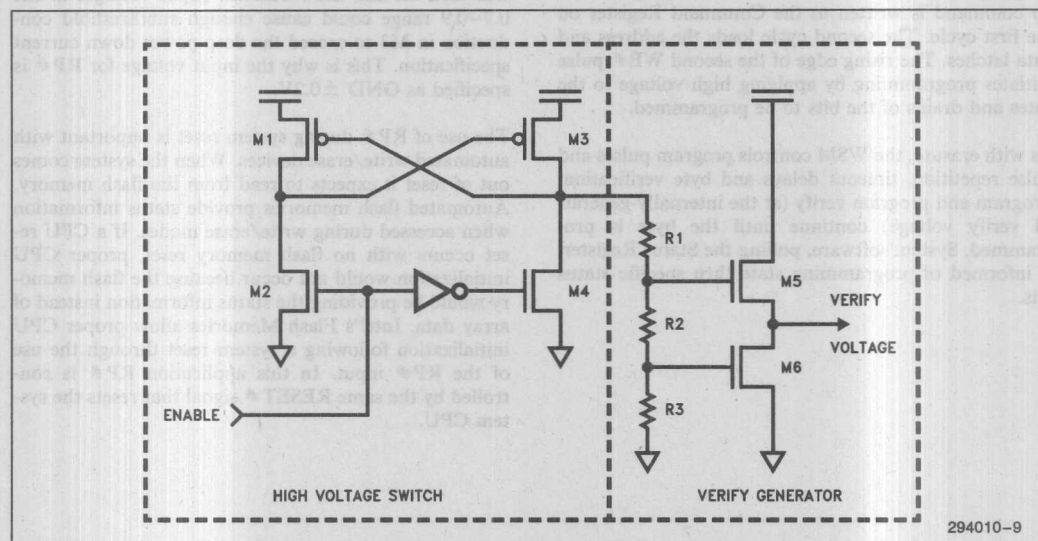


Figure 8. Erase/Program Verify Generator

DEVICE RELIABILITY

Cell Margining

Erase and program verification ensure the data retention of the newly altered memory bits. The cell margining performed by the WSM during the verify phase of the automated algorithms is more reliable than historical EEPROM schemes, as margining tests the amount of charge stored on the floating gate.

Intel's 28F001BX flash memories employ a unique circuit to internally generate the erase and program verify voltages. Figure 8 shows a simplified version of the circuit. The circuit consists of a high voltage switch and the verify voltage generator. Transistors M1 through M4 constitute the high voltage switch which disconnects V_{pp} from the resistor when the device is not in the verify mode. The verify voltage generator includes a resistor divider and a buffer. Internal margin voltage generation maintains microprocessor compatibility by eliminating the need for external reference voltages.

One of the most significant aspects of 28F001BX flash memories is their capability for 100,000 erase/program cycles per block. Destructive oxide breakdown has been a limiting factor in extended cycling of thin oxide EEPROMs. Intel's ETOX flash memory technology extends cycling performance through:

- Improved tunnel oxide processing that increases charge carrying capability tenfold;
- Reduced oxide area under stress minimizing probability of oxide defects in the region; and
- Reduced oxide stress due to a lower peak electric field (lower erase voltage than EEPROM).

A typical cell erase/program margin (V_t) is shown as a function of reprogramming cycles in Figure 9. After 100,000 reprogramming cycles, a 2.5V program read margin exists, ensuring reliable data retention.

rection of the erase V_t maximum and maintenance of a tight V_t distribution. The maximum erased V_t is set to 3.2V via the internal erase algorithm and erase verify circuits. Superior oxide quality gives an erased V_t distribution width that improves slightly with cycling (Figure 10). The tight erase V_t distribution gives an order of magnitude of erase time margin to the fastest erasing cell.

SUMMARY

Intel's ETOX flash memory technology is a breakthrough in adding electrical chip-erase to high-density EPROM technology. Intel's 28F001BX family enhances Intel's standard flash memory line by adding block erase capability, Write State Machine-controlled program and erase and deep powerdown mode. Micro-processor-compatible specifications, straightforward interfacing and in-circuit selective alterability using simple software command sequences allow designers to easily augment memory flexibility and satisfy the need for nonvolatile storage in today's designs.

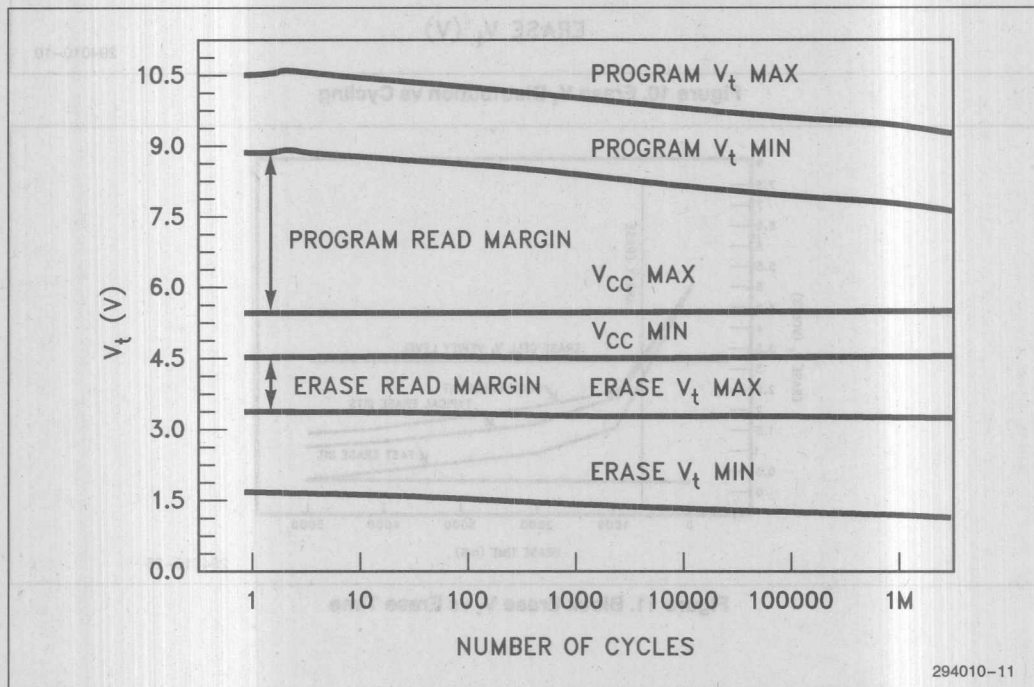
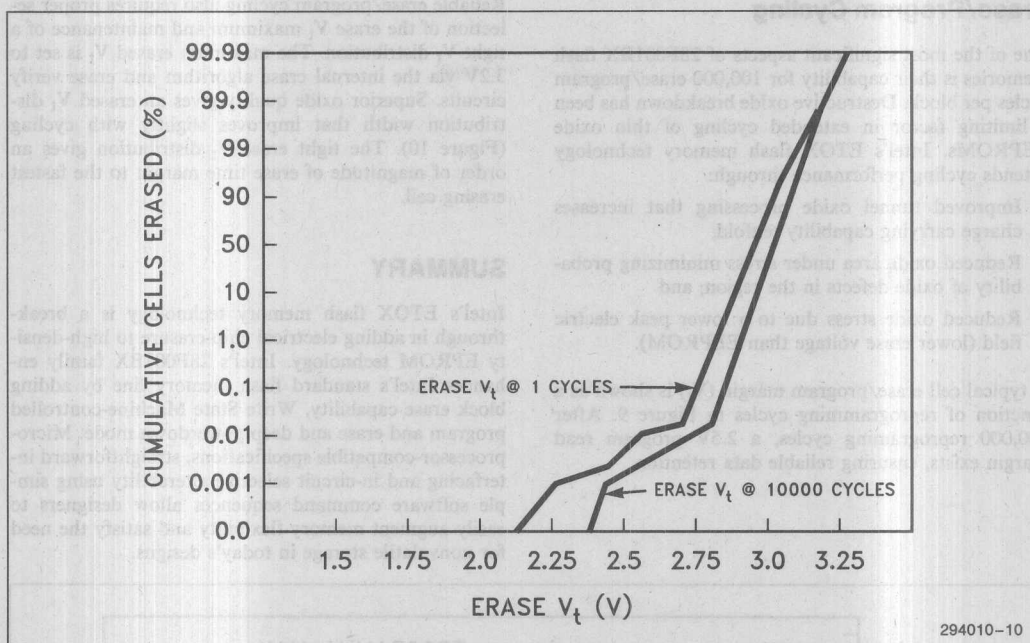
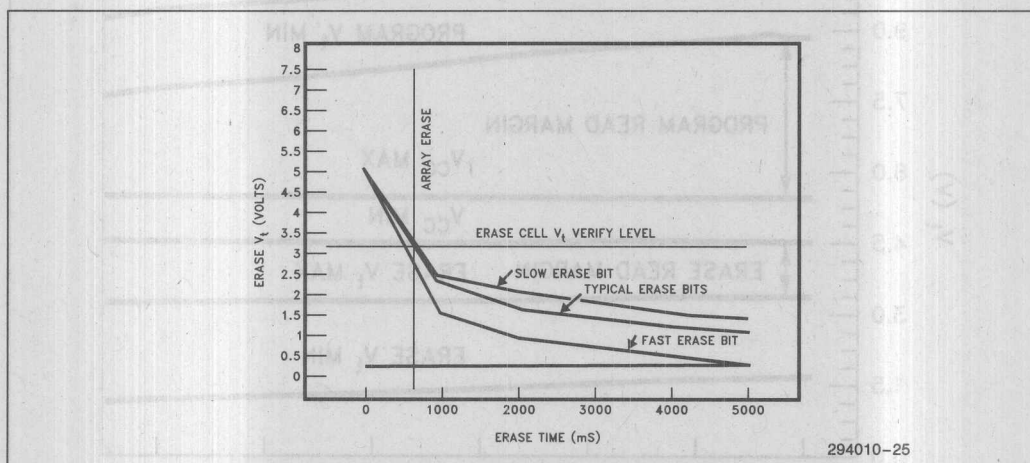
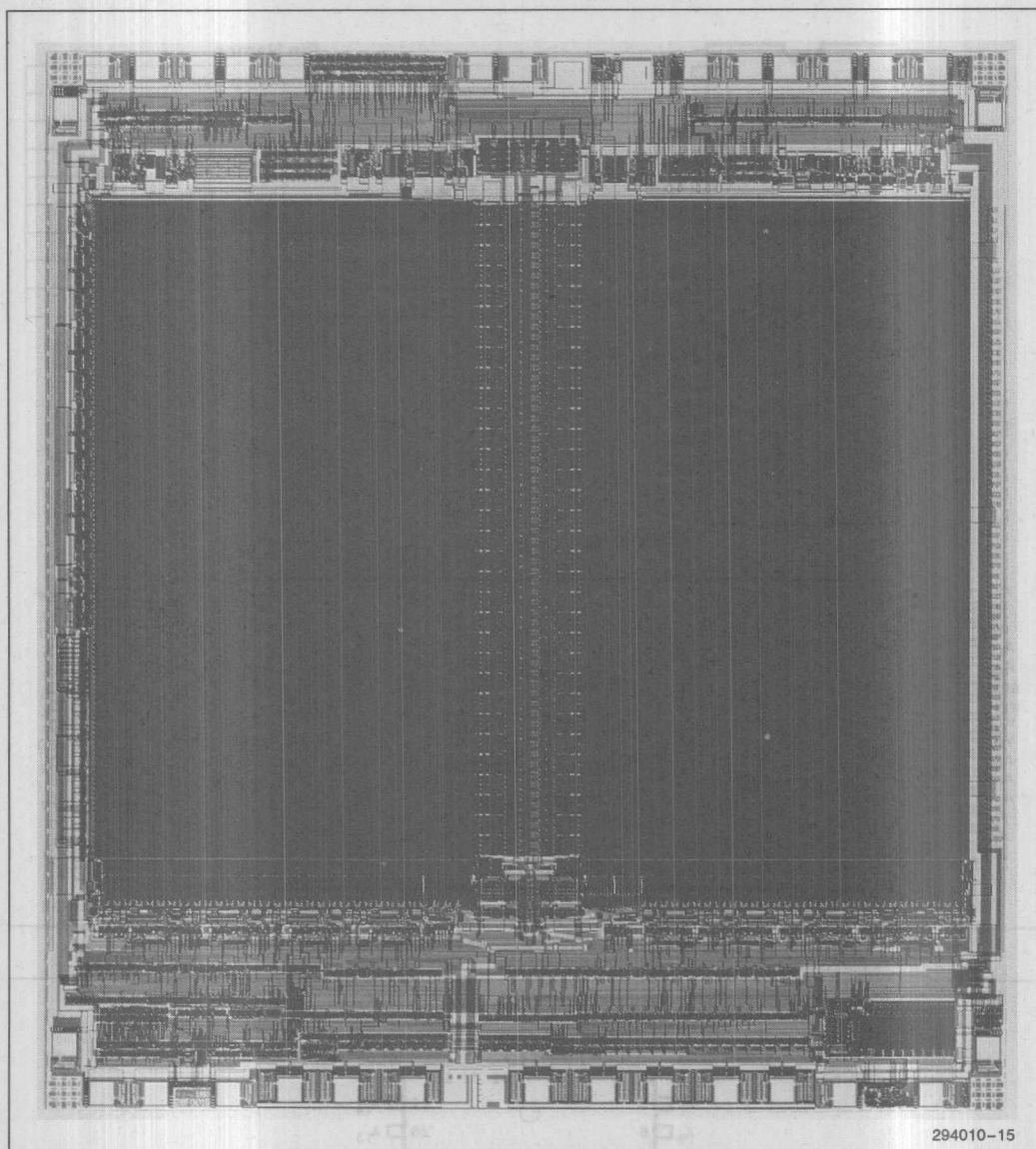


Figure 9. Block V_t vs Cycles

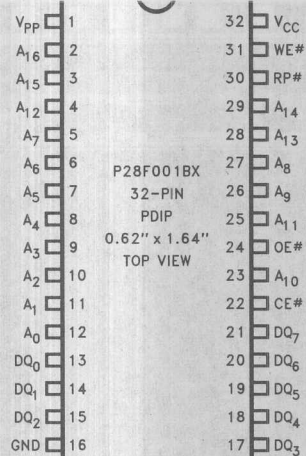
Figure 10. Erase V_t Distribution vs CyclingFigure 11. Block Erase V_t vs Erase Time



4

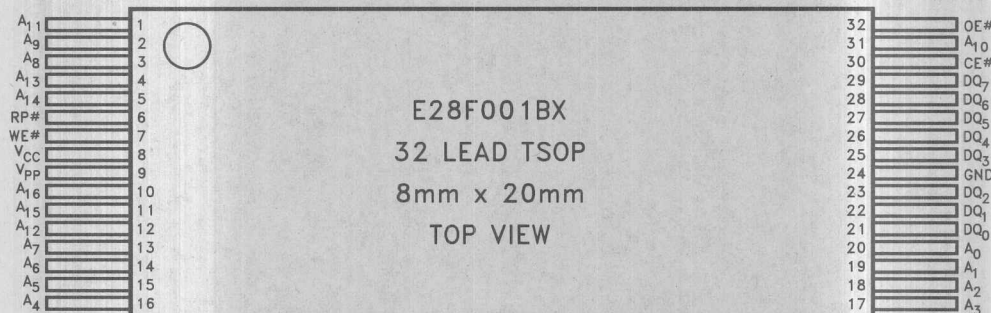
Figure 12. 28F001BX Die Photograph

294010-15



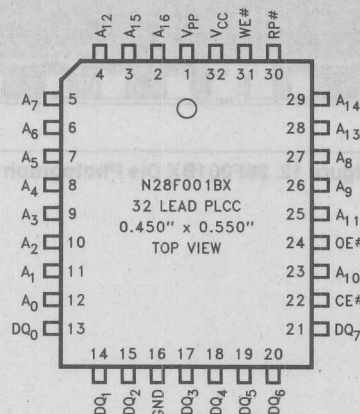
A ₀ -A ₁₆	Address Inputs
DQ ₀ -DQ ₇	Data Inputs/Outputs
CE #	Chip Enable
RP #	Power Down
OE #	Output Enable
WE #	Write Enable
V _{PP}	Program/Erase Power Supply
V _{CC}	Device Power Supply
GND	Ground

294010-12



294010-13

Figure 13. 28F001BX Pinout Configurations



294010-14

Figure 14. 28F001BX Pinout Configurations

Array Organization (Main Block):							Left Half Array IO ₀ IO ₁ IO ₂ IO ₃ BL ₅₁₁ ← BL ₆₄		Right Half Array IO ₄ IO ₅ IO ₆ IO ₇ BL ₆₄ → BL ₅₁₁		
Address							Bitlines				
A ₁₆	A ₁₅	A ₁₄	A ₁₃	A ₁₂	A ₁	A ₀	IO ₀ & IO ₇	IO ₁ & IO ₆	IO ₂ & IO ₅	IO ₃ & IO ₄	
0	0	0	0	0	0	0	BL ₄₀₀	BL ₂₈₈	BL ₁₇₆	BL ₆₄	
0	0	0	0	0	0	1	BL ₄₀₁	BL ₂₈₉	BL ₁₇₇	BL ₆₅	
0	0	0	0	0	1	0	BL ₄₀₂	BL ₂₉₀	BL ₁₇₈	BL ₆₆	
0	0	0	0	0	1	1	BL ₄₀₃	BL ₂₉₁	BL ₁₇₉	BL ₆₇	
0	0	0	0	1	0	0	BL ₄₀₄	BL ₂₉₂	BL ₁₈₀	BL ₆₈	
0	0	0	0	1	0	1	BL ₄₀₅	BL ₂₉₃	BL ₁₈₁	BL ₆₉	
0	0	0	0	1	1	0	BL ₄₀₆	BL ₂₉₄	BL ₁₈₂	BL ₇₀	
0	0	0	0	1	1	1	BL ₄₀₇	BL ₂₉₅	BL ₁₈₃	BL ₇₁	
*	*	*	*	*	*	*	*	*	*	*	
1	1	0	1	1	0	0	BL ₅₀₈	BL ₃₉₆	BL ₂₈₄	BL ₁₇₂	
1	1	0	1	1	0	1	BL ₅₀₉	BL ₃₉₇	BL ₂₈₅	BL ₁₇₃	
1	1	0	1	1	1	0	BL ₅₁₀	BL ₃₉₈	BL ₂₈₆	BL ₁₇₄	
1	1	0	1	1	1	1	BL ₅₁₁	BL ₃₉₉	BL ₂₈₇	BL ₁₇₅	

Figure 15. Bitline Decoding (Main Block, 28F001BX-T)

Array Organization (Parameter Block 1):							Right Half Array IO ₀ –IO ₇ BL ₀ → BL ₃₁							
Address							Bitlines							
A ₁₆	A ₁₅	A ₁₄	A ₁₃	A ₁₂	A ₁	A ₀	IO ₀	IO ₁	IO ₂	IO ₃	IO ₄	IO ₅	IO ₆	IO ₇
1	1	1	0	0	0	0	BL ₀	BL ₄	BL ₈	BL ₁₂	BL ₁₆	BL ₂₀	BL ₂₄	BL ₂₈
1	1	1	0	0	0	1	BL ₁	BL ₅	BL ₉	BL ₁₃	BL ₁₇	BL ₂₁	BL ₂₅	BL ₂₉
1	1	1	0	0	1	0	BL ₂	BL ₆	BL ₁₀	BL ₁₄	BL ₁₈	BL ₂₂	BL ₂₆	BL ₃₀
1	1	1	0	0	1	1	BL ₃	BL ₇	BL ₁₁	BL ₁₅	BL ₁₉	BL ₂₃	BL ₂₇	BL ₃₁

Figure 16. Bitline Decoding (Parameter Block 1, 28F001BX-T)

Array Organization (Parameter Block 2):							Right Half Array IO ₀ –IO ₇ BL ₃₂ → BL ₆₃							
Address							Bitlines							
A ₁₆	A ₁₅	A ₁₄	A ₁₃	A ₁₂	A ₁	A ₀	IO ₀	IO ₁	IO ₂	IO ₃	IO ₄	IO ₅	IO ₆	IO ₇
1	1	1	0	1	0	0	BL ₃₂	BL ₃₆	BL ₄₀	BL ₄₄	BL ₄₈	BL ₅₂	BL ₅₆	BL ₆₀
1	1	1	0	1	0	1	BL ₃₃	BL ₃₇	BL ₄₁	BL ₄₅	BL ₄₉	BL ₅₃	BL ₅₇	BL ₆₁
1	1	1	0	1	1	0	BL ₃₄	BL ₃₈	BL ₄₂	BL ₄₆	BL ₅₀	BL ₅₄	BL ₅₈	BL ₆₂
1	1	1	0	1	1	1	BL ₃₅	BL ₃₉	BL ₄₃	BL ₄₇	BL ₅₁	BL ₅₅	BL ₅₉	BL ₆₃

Figure 17. Bitline Decoding (Parameter Block 2, 28F001BX-T)

NOTES:

1. Bitline decoding listed is for 28F001BX-T. To convert to 28F001BX-B, invert polarity of addresses A₁₆–A₁₂ (i.e. 0000000 becomes 1111100).

Array Organization (Boot Block):							Left Half Array IO ₇ –IO ₀ BL ₀ → BL ₆₃							
Address							Bitlines							
A ₁₆	A ₁₅	A ₁₄	A ₁₃	A ₁₂	A ₁	A ₀	IO ₇	IO ₆	IO ₅	IO ₄	IO ₃	IO ₂	IO ₁	IO ₀
1	1	1	1	0	0	0	BL ₀	BL ₈	BL ₁₆	BL ₂₄	BL ₃₂	BL ₄₀	BL ₄₈	BL ₅₆
1	1	1	1	0	0	1	BL ₁	BL ₉	BL ₁₇	BL ₂₅	BL ₃₃	BL ₄₁	BL ₄₉	BL ₅₇
1	1	1	1	0	1	0	BL ₂	BL ₁₀	BL ₁₈	BL ₂₆	BL ₃₄	BL ₄₂	BL ₅₀	BL ₅₈
1	1	1	1	0	1	1	BL ₃	BL ₁₁	BL ₁₉	BL ₂₇	BL ₃₅	BL ₄₃	BL ₅₁	BL ₅₉
1	1	1	1	1	0	0	BL ₄	BL ₁₂	BL ₂₀	BL ₂₈	BL ₃₆	BL ₄₄	BL ₅₂	BL ₆₀
1	1	1	1	1	0	1	BL ₅	BL ₁₃	BL ₂₁	BL ₂₉	BL ₃₇	BL ₄₅	BL ₅₃	BL ₆₁
1	1	1	1	1	1	0	BL ₆	BL ₁₄	BL ₂₂	BL ₃₀	BL ₃₈	BL ₄₆	BL ₅₄	BL ₆₂
1	1	1	1	1	1	1	BL ₇	BL ₁₅	BL ₂₃	BL ₃₁	BL ₃₉	BL ₄₇	BL ₅₅	BL ₆₃

Figure 18. Bitline Decoding (Boot Block, 28F001BX-T)

NOTE:

1. Bitline decoding listed is for 28F001BX-T. To convert to 28F001BX-B, invert polarity of addresses A₁₆–A₁₂ (i.e. 0000000 becomes 1111100).

X Address										Row
A ₁₁	A ₁₀	A ₉	A ₈	A ₇	A ₆	A ₅	A ₄	A ₃	A ₂	WL
0	0	0	0	0	0	0	0	0	0	XL ₀
0	0	0	0	0	0	0	0	0	1	XL ₁
0	0	0	0	0	0	0	0	1	0	XL ₂
0	0	0	0	0	0	0	1	0	0	XL ₃
0	0	0	0	0	0	0	1	0	1	XL ₄
0	0	0	0	0	0	0	1	1	0	XL ₅
0	0	0	0	0	0	0	1	1	1	XL ₆
0	0	0	0	0	0	1	0	0	0	XL ₇
0	0	0	0	0	0	1	0	0	1	XL ₈
0	0	0	0	0	0	1	0	1	0	XL ₉
0	0	0	0	0	0	1	0	1	1	XL ₁₀
0	0	0	0	0	0	1	1	0	0	XL ₁₁
0	0	0	0	0	0	1	1	0	1	XL ₁₂
0	0	0	0	0	0	1	1	1	0	XL ₁₃
0	0	0	0	0	0	1	1	1	1	XL ₁₄
0	0	0	0	0	0	1	1	1	1	XL ₁₅
0	0	0	0	0	1	1	1	1	1	XL ₁₆
0	0	0	0	0	1	1	1	1	0	XL ₁₇
0	0	0	0	0	1	1	1	0	1	XL ₁₈
0	0	0	0	0	1	1	1	0	0	XL ₁₉
0	0	0	0	0	1	1	0	1	1	XL ₂₀
0	0	0	0	0	1	1	0	1	0	XL ₂₁
0	0	0	0	0	1	1	0	0	1	XL ₂₂
0	0	0	0	0	1	1	0	0	0	XL ₂₃
0	0	0	0	0	1	0	1	1	1	XL ₂₄
0	0	0	0	0	1	0	1	1	0	XL ₂₅
0	0	0	0	0	1	0	1	0	1	XL ₂₆
0	0	0	0	0	1	0	1	0	0	XL ₂₇
0	0	0	0	0	1	0	0	1	1	XL ₂₈
0	0	0	0	0	1	0	0	1	0	XL ₂₉
0	0	0	0	0	1	0	0	0	1	XL ₃₀
0	0	0	0	0	1	0	0	0	0	XL ₃₁

Figure 19. Wordline Decoding (28F001BX-T and 28F001BX-B)

X Address										Row
A ₁₁	A ₁₀	A ₉	A ₈	A ₇	A ₆	A ₅	A ₄	A ₃	A ₂	WL
0	0	0	0	1	0	0	0	0	0	XL ₃₂
*	*	*	*	*	*	*	*	*	*	***
0	0	0	0	1	0	1	1	1	1	XL ₄₇
0	0	0	0	1	1	1	1	1	1	XL ₄₈
*	*	*	*	*	*	*	*	*	*	***
0	0	0	0	1	1	0	0	0	0	XL ₆₃
0	0	0	1	0	0	0	0	0	0	XL ₆₄
*	*	*	*	*	*	*	*	*	*	***
0	0	0	1	0	0	1	1	1	1	XL ₇₉
0	0	0	1	0	1	1	1	1	1	XL ₈₀
*	*	*	*	*	*	*	*	*	*	***
0	0	0	1	0	1	0	0	0	0	XL ₉₅
1	1	1	1	1	0	0	0	0	0	XL ₉₉₂
*	*	*	*	*	*	*	*	*	*	***
1	1	1	1	1	0	1	1	1	1	XL ₁₀₀₇
1	1	1	1	1	1	1	1	1	1	XL ₁₀₀₈
*	*	*	*	*	*	*	*	*	*	***
1	1	1	1	1	1	0	0	0	0	XL ₁₀₂₃

Figure 19. Wordline Decoding (28F001BX-T and 28F001BX-B) (Continued)

4

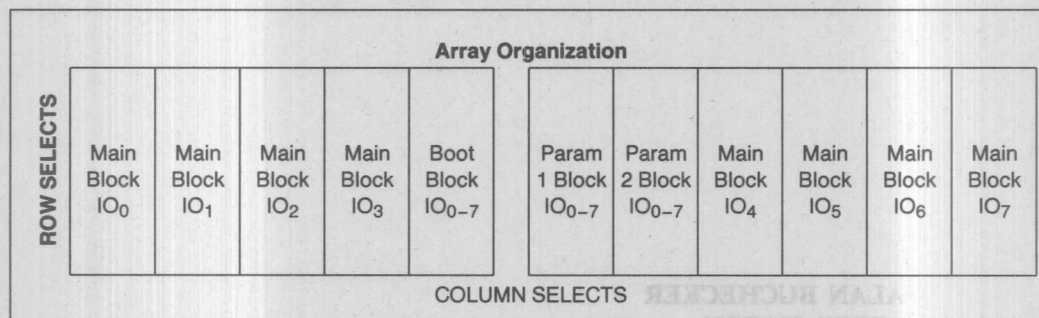


Figure 20. Array Organization

REVISION HISTORY

Number	Description
002	Swapped Figures 9 and 10 Added Figure 11
003	PWD changed to RP# for JEDEC standard compatibility.

The Intel 2/4-Mbit Boot Block Flash Memory Family

ALAN BUCHECKER
PETER HAZEN
MEMORY COMPONENTS DIVISION

October 1993

CONTENTS	PAGE
INTRODUCTION	4-290
Word/Byte-Wide Versions	4-290
Byte-Wide Versions	4-290
TECHNOLOGY OVERVIEW	4-290
DEVICE ARCHITECTURE	4-291
Array Organization	4-291
Block Memory Maps	4-292
Write/Erase Automation	4-292
Command User Interface (CUI)	4-292
Write State Machine (WSM)	4-295
Status Register	4-296
Internal Oscillator	4-297
Supply Voltage Sensing	4-297
Deep Power-Down and Device Reset	4-297
Automatic Power Savings	4-298
Block Erase	4-298
Erase Suspend/Resume	4-300
Word/Byte Write	4-300

CONTENTS	PAGE
DEVICE CHARACTERIZATION	4-300
A.C. and D.C. Parameters	4-300
Energy/Power Consumption	4-301
Word/Byte-Write and Block-Erase Times	4-301
DEVICE RELIABILITY	4-301
Word/Byte-Write and Block-Erase Cycling	4-301
Data Protection	4-302
SUMMARY	4-302
OTHER REFERENCES	4-302

INTRODUCTION

The ETOX III (EPROM tunnel oxide) 2/4-Mbit family of Intel boot block Flash Memories are a continuation of boot-top and boot-bottom architectures first introduced in the 1-Mbit 28F001BX-T and 28F001BX-B devices. Top (-T) and bottom (-B) boot block offerings provide compatibility for microprocessors that boot from high or low memory addresses.

All versions of this new family provide a 16-Kbyte hardware-protected boot block, two 8-Kbyte parameter blocks and a 96-Kbyte main block. The 2-Mbit products have an additional 128-Kbyte main block, while 4-Mbit offerings contain three additional 128-Kbyte main blocks.

Flash memories combine inherent non-volatility with in-system alterability of device contents. Selective block erasure allows manipulation of data contents within one of the seven (or five) mini-array segments without affecting data stored in the other six (or four). Advances in process control have allowed development of a double-polysilicon single-transistor flash memory capable of 100,000 write/erase cycles per block. The 2/4-Mbit boot block family electrically erases all bits in a block via electron tunneling. The EPROM programming mechanism of hot-electron injection is employed for high-performance electrical word/byte write as required for computing and embedded applications.

An integrated Command User Interface (CUI) simplifies microprocessor control of device operations (word/byte write, block erase, erase suspend/resume, Status Register read/clear, array read and device identifier read). The internal Write State Machine (WSM) controls all functions and circuits associated with word/byte-write and block-erase operations, including pulse widths and repetition, timeout delays, erase preconditioning and margined verifications. The WSM continually updates the internal Status Register during these functions. The Status Register is read via outputs DQ₀-DQ₇ providing feedback of WSM activities.

The CUI and Status Register interface to power-up/down-protection circuitry, address/data latches and the WSM. This interface augments first-generation flash memory circuitry to optimize Intel's 2/4-Mbit boot block family for microprocessor-controlled word/byte write and block erase.

A deep-power-down mode enables extremely low power consumption to augment reduced-power standby operation. An automatic power savings feature reduces I_{CC} during read mode.

A 60 ns access time (t_{ACC}) results from a memory cell current of approximately 70 μ A, low-resistance polysilicide wordlines strapped with metal, advanced scaled periphery transistors and improved circuit techniques.

The dense one-transistor cell structure, coupled with high array efficiency, yields a 4-Mbit die measuring 295 by 331 mils and a 2-Mbit die measuring 295 by 221 mils.

Word/Byte-Wide Versions

Intel's 28F400BX is a 4,194,304-bit non-volatile memory organized as either 262,144 words (256K x 16) or 524,288 bytes (512K x 8). A dedicated BYTE# control input provides selection of the desired input/output (I/O) configuration (either x16 or x8) for read operations and data writes.

Intel's 28F200BX is a 2,097,152-bit non-volatile memory organized as either 131,072 words (128K x 16) or 262,144 bytes (256K x 8). This device also provides BYTE# control of I/O configuration.

The 28F400BX and 28F200BX are available in the 44-lead Plastic Small Outline Package (PSOP) and the 56-lead Thin Small Outline Package (TSOP).

Byte-Wide Versions

Intel's 28F004BX is a 4,194,304-bit non-volatile memory arranged as 524,288 bytes (512K x 8). Intel's 28F002BX is a 2,097,152-bit non-volatile memory arranged as 262,144 bytes (256K x 8).

With lower pin counts compared to their x8/x16 equivalents, the 28F004BX and 28F002BX allow housing in the smaller 40-lead TSOP.

TECHNOLOGY OVERVIEW

Intel's ETOX III Flash Memory technology incorporates advances from ETOX I and ETOX II processes, and leverages over two decades of EPROM manufacturing experience. Using advanced 0.8 μ m double-polysilicon N-well/P-well CMOS technology, the 2/4-Mbit boot block flash memories employ a 2.5 μ m x 2.9 μ m single-transistor cell affording array density equivalent to comparable EPROM technology, and twice that of Intel's ETOX II process. The ETOX III flash memory cell is identical to 0.8 μ m EPROM, except for an additional source implant which optimizes erase performance. Figure 1 shows a cross-section of the flash memory cell.

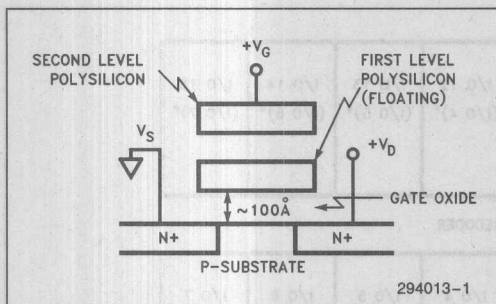


Figure 1. Flash Memory Cell

High-quality tunnel oxide under the single floating polysilicon gate promotes electrical erasure. All cells within the selected block are simultaneously erased via Fowler-Nordheim tunneling. Applying 12V to block source junctions and grounding the select gates erases all cells within that block. The internal WSM controls the automated block-erase algorithm, including pre-erase conditioning (i.e. pre-programming all block bits) and margin verification, in response to user requests relayed by the CUI. WSM-controlled block erasure, including pre-programming, typically ranges from 1.0 to 2.4 seconds depending on the size of the block selected.

Word/byte write is accomplished with the standard EPROM mechanism of channel hot-electron injection from the cell drain junction to the floating gate. Bringing both the select gate and the cell drain to high voltage initiates programming. The WSM regulates the internal word/byte-write algorithm, including margin verification, after the correct command sequence is written and decoded. Word/byte write typically requires 9 μ s.

DEVICE ARCHITECTURE

Array Organization

Layout of the 2/4-Mbit boot block family is segmented as two array planes (see Figure 2). This organization allows improved access times via minimal internal busing, thus balancing the need for high speed with the requirement of small die size for cost-effective solutions. In the 4-Mbit family, each array plane is 1024 rows by 2048 columns. For the 2-Mbit family, each array plane is 1024 rows by 1024 columns. Access time

is reduced by limiting column length to 1024 cells. The polysilicon row is strapped in metal every 512 columns to reduce wordline delay. Two row decoders run vertically along the array plane sides, and column decoders run horizontally between array planes. Figure 36 shows a die photo of the 4-Mbit family.

Each array plane is divided into eight I/Os for the 28F400BX/200BX. The upper plane consists of the high-byte I/Os (DQ₈–DQ₁₅), while the lower plane consists of the low-byte I/Os (DQ₀–DQ₇). During byte-wide operation (BYTE# = "0") the high-byte I/Os are multiplexed through the low-byte I/Os via A₋₁ decoding. Data for I/O₀ is stored in the left-most 256 columns (or 128 columns for the 28F200BX) of the lower plane, with the next 256 columns (or 128) storing data for I/O₁, etc. Data for I/O₈ is stored in the left-most 256 (or 128) columns of the upper plane, with the next 256 (or 128) columns storing data for I/O₉, etc.

For the dedicated byte-wide products (28F004BX/002BX), data for a given I/O is divided between the upper and lower array planes as decoded by A₁₀. Data for I/O₀ is stored in the left-most 256 columns (or 128 columns for the 28F002BX) of both the upper and lower planes, with the next 256 columns (or 128) in each plane storing data for I/O₁, etc.

Since each I/O is a grouping of adjacent columns (256 or 128), the independently-erasable blocks (seven or five) are segmented within each I/O. Each I/O in the 4-Mbit family is divided into seven blocks; including a 16-Kbyte boot block, two 8-Kbyte parameter blocks, one 96-Kbyte main block and three 128-Kbyte main blocks. Each I/O in the 2-Mbit family is divided into five blocks; including a 16-Kbyte boot block, two 8-Kbyte parameter blocks, one 96-Kbyte main block and one 128-Kbyte main block. Each block source is electrically isolated from the sources of the other six (or four) blocks. This allows individual block erase without altering data in the other blocks.

Addresses A₉–A₀ select one of 1024 rows. Row address lines are decoded sequentially for selection. Table 3 lists row address bitmaps.

Columns are decoded by A₁₈–A₁₀ for the 28F004BX, A₁₇–A₁₀ for the 28F400BX and 28F002BX, and A₁₆–A₁₀ for the 28F200BX. 4-Mbit family columns are numbered 0–255 for each I/O, and 2-Mbit columns are numbered 0–127 for each I/O. Table 4 and 5 lists column address bitmaps.

4

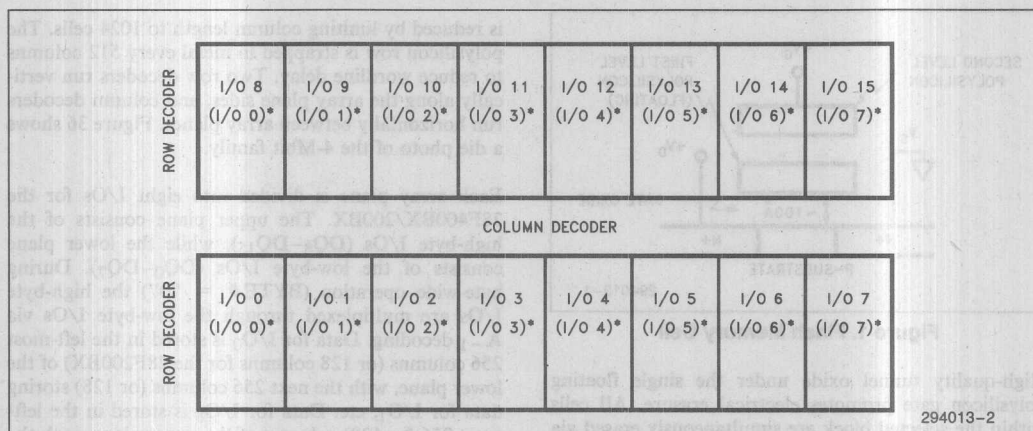


Figure 2. 2/4-Mbit Boot Block Family Array Organization ((I/O#)* Denotes 28F004BX/002BX I/Os)

Block Memory Map

Top (-T) and bottom (-B) boot block offerings have block address maps which are inverted from one another to provide compatibility for microprocessors that boot from high or low memory addresses. Figures 3, 4, 5 and 6 show 28F400BX-T/B, 28F200BX-T/B, 28F004BX-T/B and 28F002BX-T/B memory maps. The addresses shown in Figures 3 and 4 for the 28F400BX-T/B and 28F200BX-T/B, decode two bytes of data. A₁ decodes between the two bytes when BYTE# is low.

Write/Erase Automation

Intel's 2/4-Mbit boot block Flash Memories contain an on-chip Command User Interface. Write State Machine, Status Register and address/data latches to dramatically simplify user interface. This combination of functional units reduces microprocessor control

complexity of word/byte-write block-erase, erase-suspend/resume, Status Register-read/clear, ID-read and array-read operations. Figures 7 and 8 show 28F400BX/200BX and 28F004BX/002BX block diagrams.

Command User Interface (CUI)

The CUI consists of a command decoder and command register. User requests are decoded and latched in a microprocessor write cycle controlled by Chip Enable (CE#) and Write Enable (WE#). Status Register-read/clear, ID-read and array-read commands are directly handled by the CUI. The CUI also accepts word/byte-write, block-erase and erase-suspend/resume commands. WE#'s rising edge latches address, command and data-in registers, and requests WSM initiation of the selected operation. These on-chip address, command and data latches, controlled by the CUI, minimize system interface logic and free the system bus.

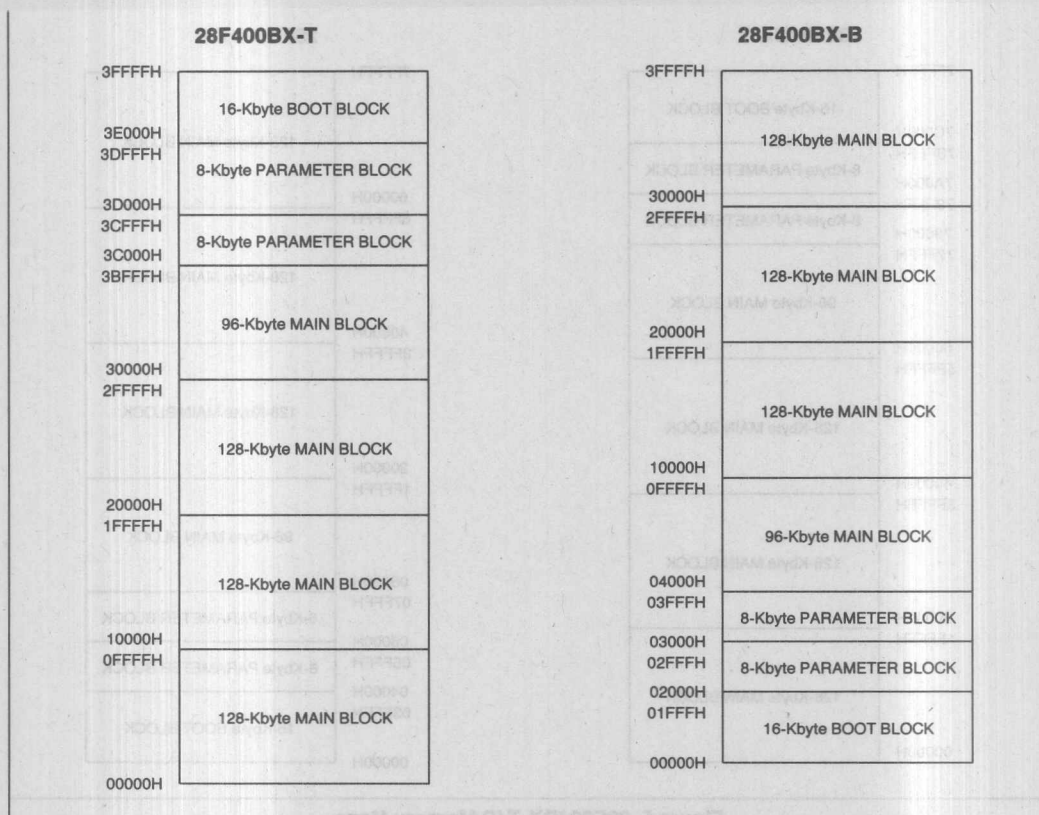


Figure 3. 28F400BX-T/B Memory Maps

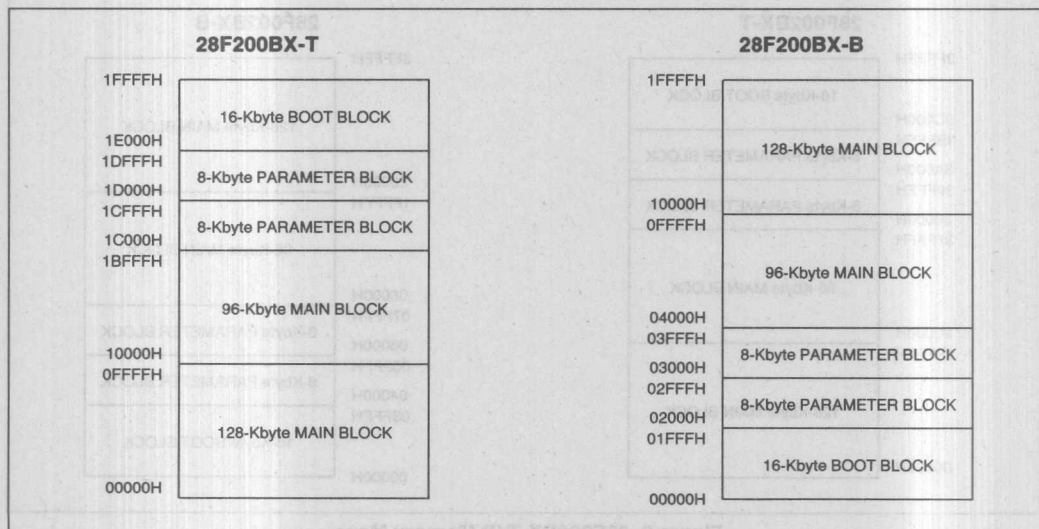


Figure 4. 28F200BX-T/B Memory Maps

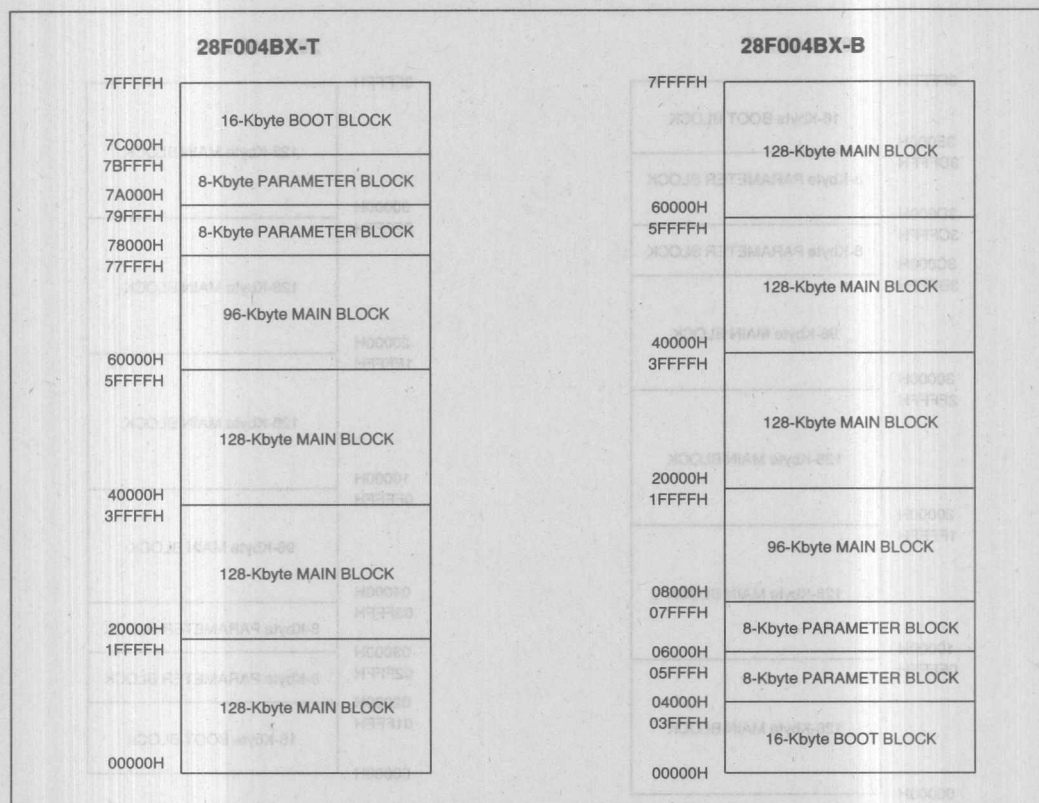


Figure 5. 28F004BX-T/B Memory Maps

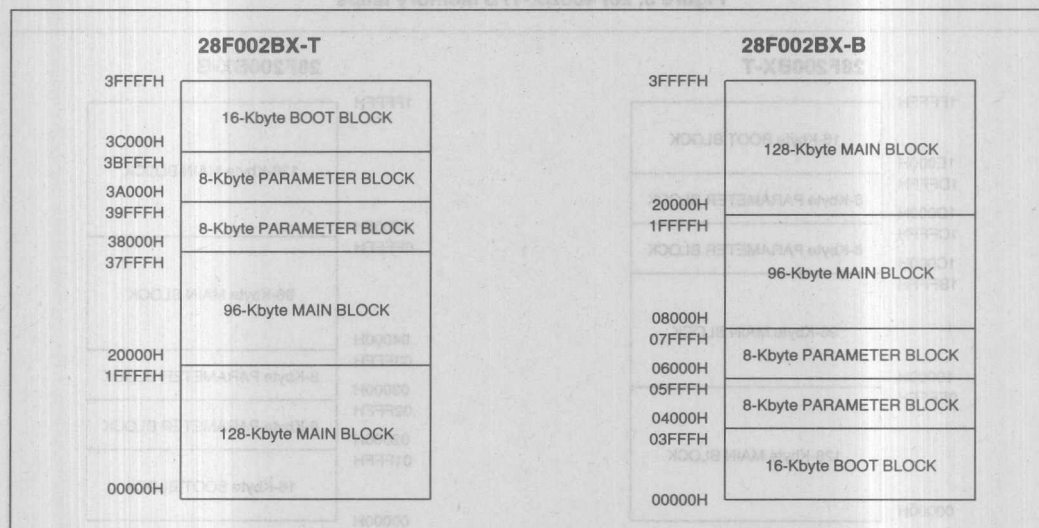


Figure 6. 28F002BX-T/B Memory Maps

Write State Machine (WSM)

The WSM processes word/byte-write, block-erase and erase-suspend/resume requests received from the CUI. The WSM rejects word/byte-write and block-erase requests if the WSM is currently busy, if V_{PP} is not at high voltage (12V) or if the Low V_{PP} Status Register flag is set (i.e. not cleared from a previous low-voltage condition).

The WSM consists of an integrated oscillator and control circuitry. It generates signals which control the word/byte-write, block-erase, erase-suspend/resume and verify circuits. It also receives feedback from these circuits, allowing Status Register updates. The WSM and associated circuits perform the equivalent of first-generation flash memory program and bulk-erase algorithms automatically.

This eliminates the need for system timers, and frees the microprocessor to service interrupts or perform other functions during device word/byte-write or block-erase operations.

The WSM provides feedback to the CUI to determine when a given command is valid. Although nearly all commands are available when the WSM is inactive, only status read is valid while the WSM performs a word/byte-write operation. During block erase, only the read-status and erase-suspend commands are available. Read-array, read-status, and erase-resume commands are valid with the WSM in an erase-suspended state. Invalid operations are interpreted as the read-array command when the WSM is inactive or erase suspended, and as the read-status command when the WSM is active in word/byte write or block erase.

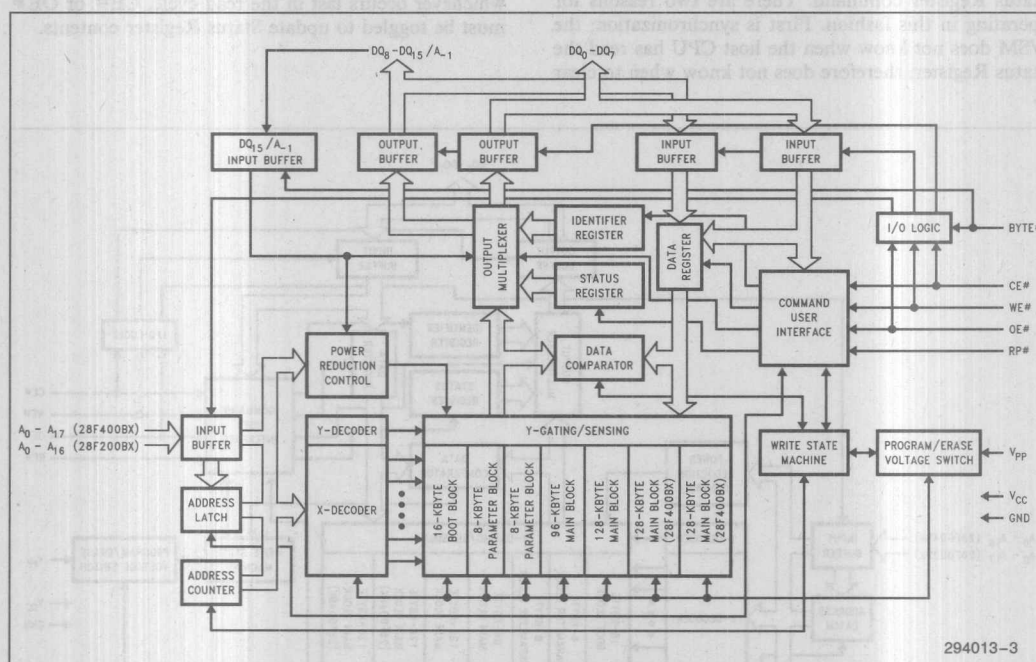


Figure 7. 28F400BX/200BX Block Diagram

Status Register

The internal Status Register contains a full complement of activity status bits. These bits and their meaning, "1/0" are:

SR.7: WSM status (READY/BUSY#)

SR.6: Erase-suspend status (ERASE SUSPENDED/
ERASE IN PROGRESS# or
COMPLETED#)

SR.5: Block-erase status (ERROR/SUCCESS#)

SR.4: Word/byte-write status (ERROR/SUCCESS#)

SR.3: V_{pp} status (LOW/OK#)

All bits are set by the WSM, and read via the CUI. The WSM can only set SR.3, SR.4 and SR.5; it cannot clear them. They remain set until the CUI processes a clear Status Register command. There are two reasons for operating in this fashion. First is synchronization; the WSM does not know when the host CPU has read the Status Register, therefore does not know when to clear it.

Secondly, allowing system software to control reset adds flexibility to the way this device may be used. The CPU may write several words/bytes, or erase several blocks back-to-back, while polling SR.7 to determine when the next word/byte-write or block-erase command can be given. When all words/bytes are written, or all blocks erased, the system polls the other status flags to determine if all operations were successful, or if an error occurred. While other approaches require the controlling microprocessor to watch for non-completion of write or erase within a specified time to indicate an error, this implementation requires no external system timers or software timing loops. As such, the system can reduce its polling overhead while still identifying any potential error conditions.

Status Register contents are driven to device outputs on the falling edge of CE# or Output Enable (OE#), whichever occurs last in the read cycle. CE# or OE# must be toggled to update Status Register contents.

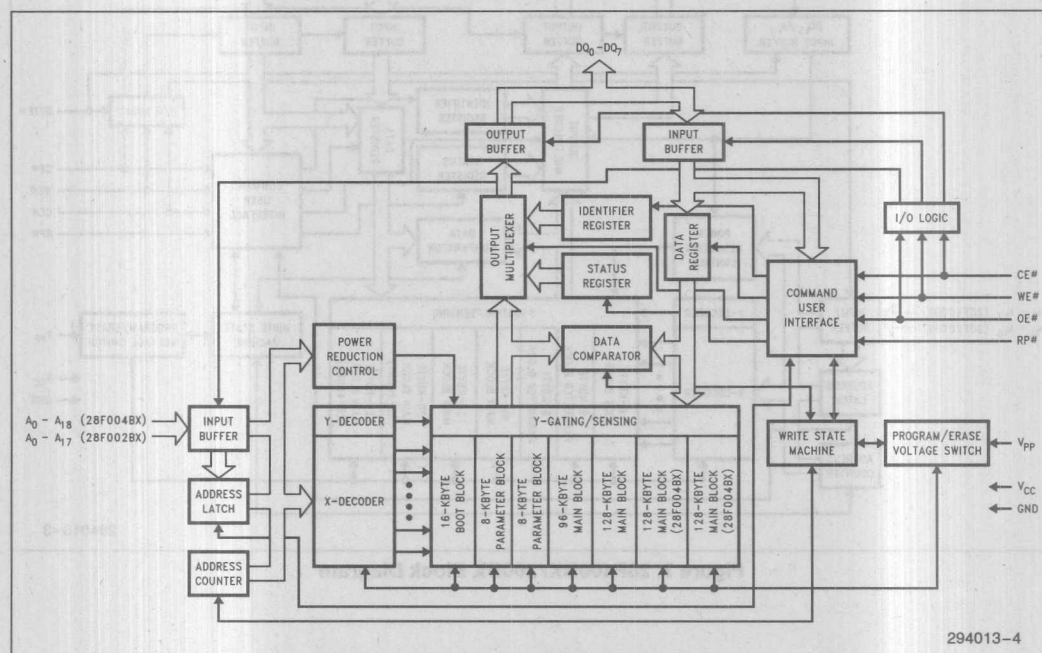


Figure 8. 28F004BX/002BX Block Diagram

Internal Oscillator

The WSM is designed using clocked logic circuits. An on-chip ring oscillator generates the clock signals. The frequency of a standard ring oscillator varies with processing, temperature and supply voltage. A proven design used on the 28F001BX-T/B, 28F008SA and 2/4-Mbit boot block family minimizes these variations.

The switching current of each stage in the ring oscillator is controlled by a current reference which varies linearly with V_{CC} . The trip point of each ring oscillator inverter also varies linearly with V_{CC} . These two effects offset each other, and the resulting oscillator period is proportional to RC with only a small dependence on V_{CC} .

An on-chip resistor sets the value of R. The gate capacitance of the inverters in the ring oscillator sets the value of C. Process variations in these values are reduced by trimming the period of each oscillator during manufacturing. The resistor is the only source of temperature variation.

Supply Voltage Sensing

Figure 9 shows the $LOWV_{CC}$ and $LOWV_{PP}$ generation circuits. Power supply voltages (V_{CC} and V_{PP}) are divided down and compared to a reference voltage. If V_{REF} is greater than the divided power supply voltage, the $LOWV_{CC}$ or $LOWV_{PP}$ signal is driven high. The generated V_{REF} level is supply-voltage independent to the first order.

Positive power to the circuit is supplied by M1 and M2. M1 and M2 sources are pulled up to the higher of ($V_{PP} - V_{tn}$) or ($V_{CC} - V_{tw}$). V_{tn} is the threshold of an implanted N-channel device, about 0.9V. V_{tw} is the threshold of a native N-channel device, about 0V. This scheme ensures that the circuit works regardless of the applied supply voltages.

The $LOWV_{CC}$ signal is used by the word/byte-write and block-erase circuits, as well as the CUI and WSM.

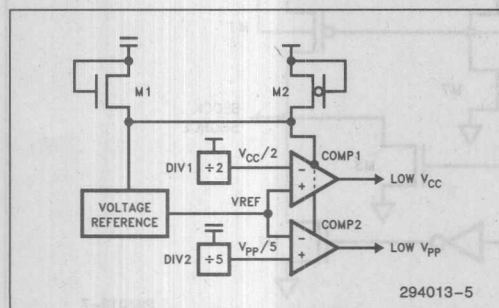


Figure 9. Low-Power Detector Circuit

If $LOWV_{CC}$ is active, the CUI will not accept user writes and resets to an array-read condition. The WSM is similarly reset by $LOWV_{CC}$. The $LOWV_{PP}$ signal is used by the WSM; if V_{PP} drops below the high-voltage detector trip point during word/byte write or block erase, the Status Register's low V_{PP} bit is set and WSM operation halts. The system must clear the Status Register before any subsequent word/byte-write or block-erase operations can succeed.

Deep Power-Down and Device Reset

The 2/4-Mbit boot block family incorporates a deep-power-down mode that reduces I_{CC} and I_{PP} to typically 0.20 μA and 0.07 μA respectively. $RP\#$ low selects deep power-down. When $RP\#$ is high, the device can be placed in an active or standby mode depending on $CE\#$ state.

Deep power-down is similar to standby except that all circuits excluding the $RP\#$ buffer are turned off. This mode greatly reduces power consumption, but requires more time to transition the device into an active mode. A read wake-up time (t_{PHQV}) is required from $RP\#$ switching high until output and sense circuitry become fully functional and data can be read from the part. Similarly, a write wake-up time (t_{PHWL}) is needed before the CUI recognizes writes. After this interval normal operation is restored; the CUI is reset to read-array mode and the Status Register is cleared to 80H.

Figure 10 shows a diagram of the power-down circuit. The TTL buffer formed by M1–M3 disables the low-power detect circuits, the redundancy-address flash bits and the $CE\#$ TTL buffer formed by M4–M6. These circuits were always enabled in first-generation Intel flash architectures. Turn-on delays of these circuits determine $RP\#$ access time and write specifications.

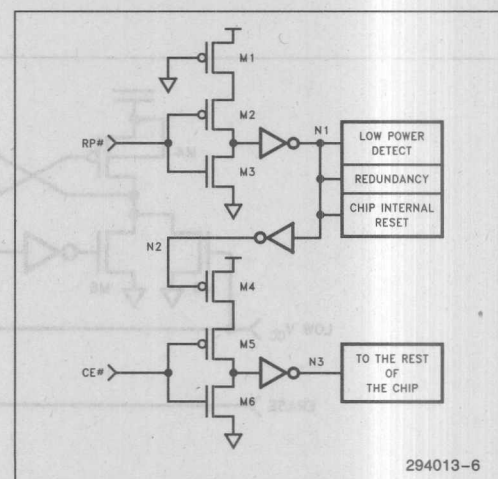


Figure 10. Power-Down and Reset Functions

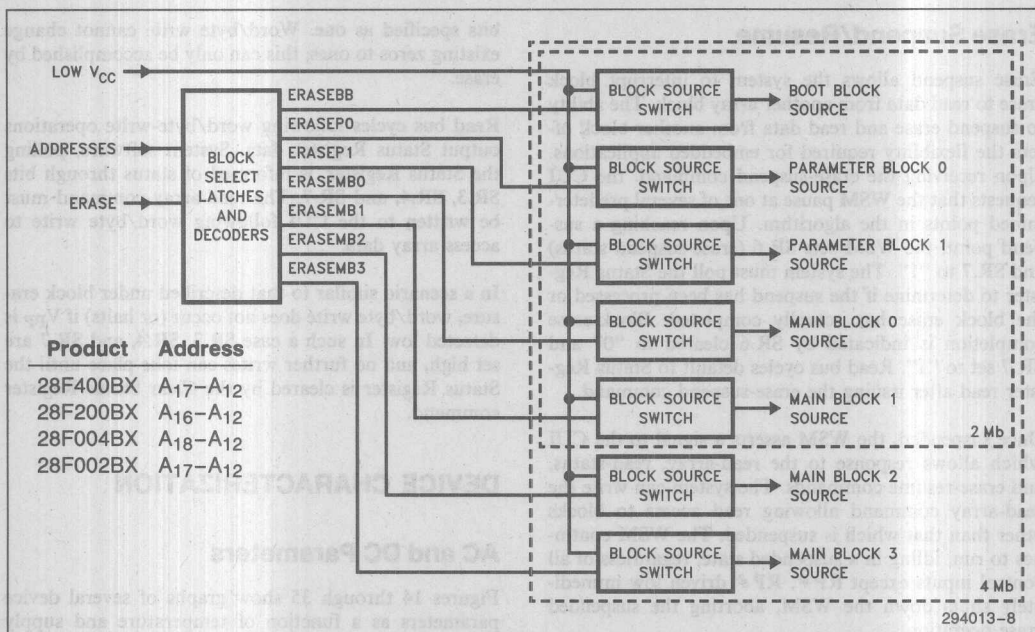


Figure 12. Array Erase Blocking

The tunneling that occurs during block erase requires only a small amount of current. However, the initial current required to charge the block's cumulative source capacitance to the erase voltage is large. Supply decoupling design practices minimize the system impact of the source charging.

The LOWV_{CC} signal protects the array from erasure when V_{pp} is at a high voltage, but V_{CC} is below the write/erase lockout voltage (V_{LKO}). When this occurs, M3 pulls the block source to ground. The high-voltage latch is forced by M8 into the state that turns M1 off.

V_{pp} is continually monitored during all phases of the block-erase operation. If V_{pp} falls below the trip point of its high-voltage detect circuitry, erasure will not occur (or halts) and Status Register V_{pp} status (SR.3), block-erase status (SR.5) and WSM status (SR.7) bits are set to "1".

If SR.3 is set (Low V_{pp}), WSM operation is inhibited. The WSM will not execute further word/byte-write or block-erase sequences until the Status Register has been reset by system software. Word/byte-write or block-erase requests with error flags SR.4 or SR.5 set are not inhibited, but the system loses the ability to determine success. The clear Status-Register command resets these bits.

After receiving the block-erase command sequence, the WSM automatically controls block pre-condition (programming all words to 0000H within the chosen block), erase pulses and pulse repetition, timeout delays, and word-by-word verification of all block addresses (sequentially checked via the address counter) using an alternative sensing reference to verify margin. The internal erase and verify operations continue until the entire block is erased. A read cycle applied to the part following the block-erase command sequence outputs Status Register contents; system software can poll the Status Register to determine when block erase has successfully completed. Following block erasure, the device remains in Status Register read block erasure; the device remains in Status Register read mode; a read-array command must be written to the device to access array data.

If the erase-setup command is followed with a command other than erase confirm, the device will not erase. The WSM sets both word/byte-write status and block-erase status bits in the Status Register to indicate an invalid sequence.

Erase Suspend/Resume

Erase suspend allows the system to interrupt block erase to read data from another array block. The ability to suspend erase and read data from another block offers the flexibility required for embedded applications. Upon receiving the erase-suspend command, the CUI requests that the WSM pause at one of several predetermined points in the algorithm. Upon reaching a suspend point, the WSM sets SR.6 (erase-suspend status) and SR.7 to "1". The system must poll the Status Register to determine if the suspend has been processed or the block erase has actually completed. Block-erase completion is indicated by SR.6 cleared to "0" and SR.7 set to "1". Read bus cycles default to Status Register read after issuing the erase-suspend command.

Once suspended, the WSM asserts a signal to the CUI which allows response to the read-array, read-status, and erase-resume commands. The system can write the read-array command allowing read access to blocks other than that which is suspended. The WSM continues to run, idling in a suspended state, regardless of all control inputs except RP#. RP# driven low immediately shuts down the WSM, aborting the suspended erase operation.

The erase-resume command must be issued upon completion of reads from other array blocks to continue block-erase operation. The WSM then clears SR.6 and SR.7, and resumes erase operation from the suspension point. Read cycles following the erase-resume command output Status Register data.

Word/Byte Write

Word/byte write follows a flow similar to block erase. The word/byte-write-setup command is first written to the CUI. A second write cycle loads address and data latches. The rising edge of the second WE# pulse requests that the WSM initiate activity, applying high voltage to the gates and drains of all bits to be written. Unlike block erase, word/byte write will proceed regardless of what data is applied on the second CUI write cycle; however writing data FFFFH (or FFH) does not modify memory contents.

Like block erase, the WSM controls program pulses and pulse repetition, timeout delays and word/byte verification. Word/byte write and verify (with alternate sensing reference and internally-generated verify voltage) continue until the word/byte is written. Internal word/byte-write verify checks that all bits written to zero have been correctly modified; it does not check

bits specified as one. Word/byte write cannot change existing zeros to ones; this can only be accomplished by erase.

Read bus cycles following word/byte-write operations output Status Register data. System software, polling the Status Register, is informed of status through bits SR.3, SR.4, and SR.7. The read-array command must be written to the CUI following word/byte write to access array data.

In a scenario similar to that described under block erase, word/byte write does not occur (or halts) if V_{pp} is detected low. In such a case SR.3, SR.4, and SR.7 are set high, and no further writes can take place until the Status Register is cleared by the Clear Status Register command.

DEVICE CHARACTERIZATION

AC and DC Parameters

Figures 14 through 35 show graphs of several device parameters as a function of temperature and supply voltage.

In particular, note Figure 14 which shows typical read performance t_{AVQV} (t_{ACC}) of the 28F400BX, which is representative of the 2/4-MBit boot block flash memory family, as a function of V_{CC} and ambient temperature. t_{ELQV} (t_{CE}) in Figure 15 and t_{GLQV} (t_{OE}) in Figure 16 are also of particular interest. Access times t_{AVQV} , t_{ELQV} , and t_{GLQV} are specified and tested with an output load of 100 pF; decreased output load capacitance improves device operation.

Table 1 shows typical supply currents for several operating modes.

Table 1. RMS Current Values

Mode	I_{CC} ($V_{CC} = 5.0V$, CMOS Inputs)		I_{PP} ($V_{PP} = 12V$)
	x8	x16	
Read (6 MHz)	23 mA	25 mA	100 μA
Write	20 mA	22 mA	10 mA
Block Erase	15 mA	15 mA	12 mA
Standby	50 μA	50 μA	100 μA
Deep Power-Down	0.20 μA	0.20 μA	0.07 μA

Energy/Power Consumption

The system designer may be concerned with power consumption during block erase and word/byte write. Figure 32 shows I_{CC} and I_{pp} during block erase. Figure 33 shows I_{CC} and I_{pp} during word/byte write.

Word/Byte-Write and Block-Erase Times

The 2/4-Mbit boot block family and 28F008SA advance word/byte-write and block-erase performance compared to previous flash memories. The on-chip algorithm is improved over the 28F001BX to take advantage of process enhancements. This improvement is most apparent when compared to first-generation flash memory parts with externally controlled algorithms. First-generation device times shown in Table 2 assume optimal system overhead, and as such are absolute best case.

Table 2. Word/Byte-Write and Block-Erase Performance vs Previous Devices

Device	Word/Byte-Write Time	Block-Erase Time/ # Bytes	Erase Time per Kbyte
SECOND-GENERATION FLASH MEMORY DEVICES ⁽¹⁾			
28F400BX	9 μ s	2.4s/128KB	19 ms
28F004BX	9 μ s	2.2s/96KB	23 ms
28F200BX	9 μ s	1.0s/16KB	63 ms
28F002BX	9 μ s	1.0s/8KB	125 ms
28F008SA	9 μ s	1.5s/64KB	23 ms
28F001BX	18 μ s	3.8s/112KB	34 ms
		2.1s/8KB	256 ms
		2.1s/4KB	513 ms
FIRST-GENERATION FLASH MEMORY DEVICES ⁽²⁾			
28F020	16.5 μ s	6.8s/256KB	27 ms
28F010	16.5 μ s	3.9s/128KB	30 ms
28F512	16.5 μ s	2.4s/64KB	37 ms
28F256A	16.5 μ s	1.6s/32KB	51 ms

NOTES:

1. Typical measured time.

2. Times calculated based on typical erase and precondition pulse requirements, with minimum write timings. Calculations are described in Figure 13.

DEVICE RELIABILITY

Word/Byte-Write and Block-Erase Cycling

One of the most important reliability aspects of the 2/4-Mbit boot block family is its capability of 100,000 write/erase cycles per block. Destructive oxide breakdown has been a limiting factor in extended cycling of thin-oxide EEPROMs. Intel's ETOX Flash Memory technology extends cycling performance through:

- Improved tunnel-oxide processing that increases charge-carrying capability tenfold.
- Significantly reduced oxide area under stress that minimizes probability of oxide defects in the region.
- Reduced oxide stress due to a lower peak electric field (lower erase voltage than EEPROM).

reliable word/byte-write and block-erase cycling requires proper selection of the maximum erase threshold voltage (V_t), and maintenance of a tight distribution. Maximum erase V_t is set to 3.4V via the internal block-erase algorithm and verify circuits. Tight erase V_t distribution gives an order of magnitude of erase-time margin to the fastest erasing cell, with virtually identical erase V_t distributions at 1 and 10,000 cycles (Figure 34). Program V_t distribution is similarly consistent over cycling (Figure 35).

Data Protection

The 2/4-Mbit boot block family offers protection against accidental block erasure or word/byte write during power transitions. Internal circuitry creates a device insensitive to V_{PP}/V_{CC} supply power-up/down sequencing.

$V_{PP} \leq V_{PPLMAX}$ locks out word/byte-write and block-erase circuits. $V_{CC} \leq V_{LKO}$ disables CUI command writes, resets the CUI to array-read mode, and holds the WSM inactive. The system designer must still guard against spurious command writes for $V_{CC} > V_{LKO}$ when $V_{PP} < V_{PPLMAX}$.

Several strategies are available to prevent data modification in the 2/4-Mbit family. The CUI provides a degree of software write protection since memory alteration occurs only after successful completion of a two-step write sequence. $WE\#$ and $CE\#$ must both go active to perform this sequence; driving either high inhibits command/data writes. Secondly, the system can place the device in deep-power-down mode ($RP\# = V_{IL}$) to disable command writes, reset the CUI to array-read mode, and hold the WSM inactive, effectively protecting array data and providing a way to reset the flash memory during system reset conditions. Finally, the system designer may switch V_{PP} to V_{PPH} when memory updates are required.

SUMMARY

Intel's 2/4-Mbit boot block Flash Memory family contains features that optimize this product group for computing and embedded applications. These features include hardware-implemented write/erase protection of the boot block, specialized array blocking, dedicated x8 and x8/16 user-configurable versions, automation of word/byte write and block erase, erase suspend for data read, deep-power-down/automatic-power-savings modes, reset capability, and a write/erase Status Register. With simple microprocessor interfacing and software command sequences, this family is the non-volatile computing and embedded solution of choice for today's designs.

OTHER REFERENCES

Related documents of interest to readers of this engineering report:

28F400BX-T/B, 28F004BX-T/B; "4-Mbit Flash Memory Family" Data Sheet (Order #290450)

28F400BX-TL/BL, 28F004BX-TL/BL; "4-Mbit Flash Memory Family" Data Sheet (Order #290451)

28F200BX-T/B, 28F002BX-T/B; "2-Mbit Flash Memory Family" Data Sheet (Order #290448)

28F200BX-TL/BL, 28F002BX-TL/BL; "2-Mbit Flash Memory Family" Data Sheet (Order #290449)

ER-28 "ETOX III Flash Memory Technology" Engineering Report (Order #294012)

AP-341 "Designing an Updatable BIOS Using Flash Memory" (Order #292077)

AP-363 "Extended Flash BIOS for Portable Computers" (Order #292098)

SUPPLEMENTARY INFORMATION

FORMULA:

b = # bytes in a block (256K, 128K, 64K, 32K)
 n = # of erase pulses required (90 pulses)
 w = Time for a write cycle (150 ns, t_{AVAV})
 v = Time to verify (6055 ns, $t_{WHGL} + t_{GLQV}$)
 p = Program pulse width (10 μ s, t_{WHWH1})
 one pulse programming assumed
 e = Erase pulse width (10 ms, t_{WHWH2})

Precondition and precondition verify time is:
 $b(2w + p + v)$

Erase/verify, each loop where some byte does not pass verify:
 $(n - 1)(2w + e + v)$

Last erase pulse:
 $(1)(2w + e)$

Passing erase-verify, all bytes:
 $b(w + v)$

Total time can be summarized as:
 $b(3w + p + 2v) + n(2w + e + v) - (v)$

or substituting in times for write, verify, program and erase pulse widths:
 $b(22.56 \times 10^{-6}) + n(10.006355 \times 10^{-3}) - (6.055 \times 10^{-6})$ Seconds

Figure 13. Erase Time Calculations for First-Generation Flash Memories

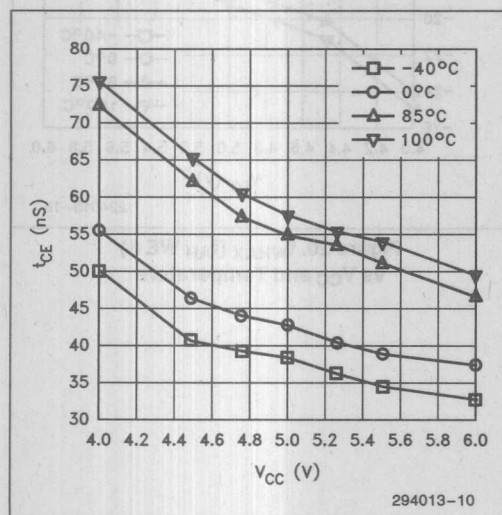


Figure 15. t_{ELQV} (t_{CE}) vs V_{CC} and Temperature

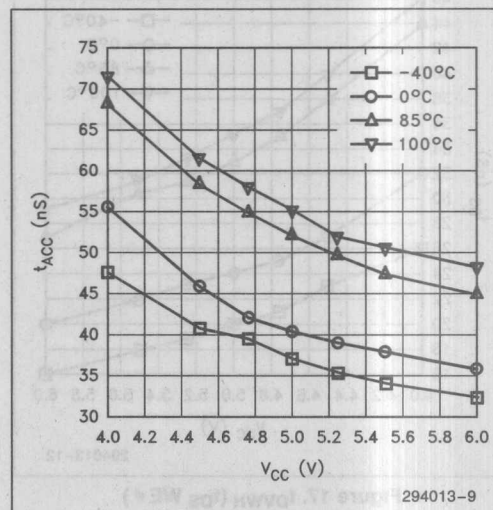


Figure 14. t_{AVQV} (t_{ACC}) vs V_{CC} and Temperature

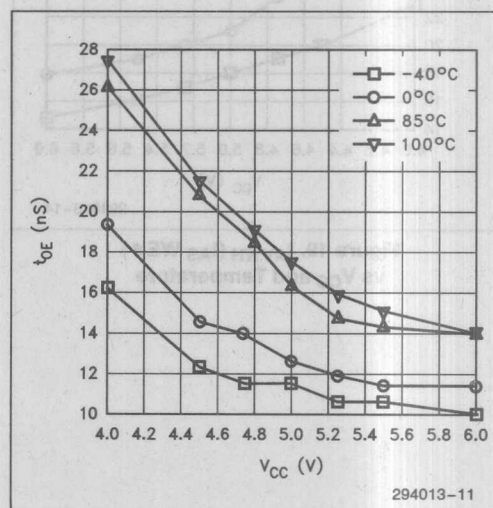


Figure 16. t_{GLQV} (t_{OE}) vs V_{CC} and Temperature

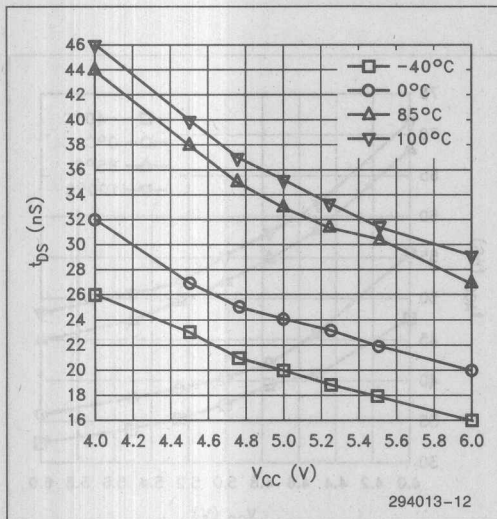


Figure 17. t_{DVWH} (t_{DS} WE#)
vs V_{CC} and Temperature

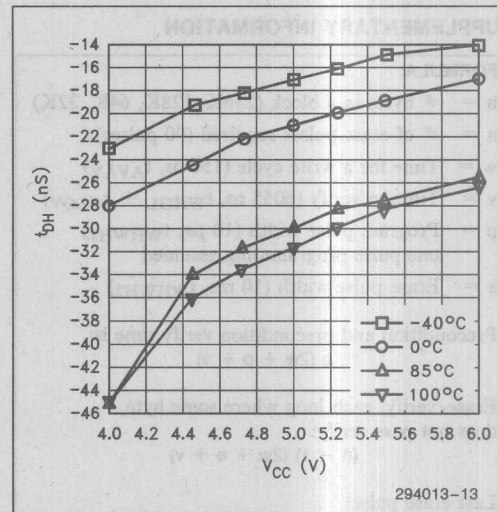


Figure 18. t_{WHDH} (t_{DH} WE#)
vs V_{CC} and Temperature

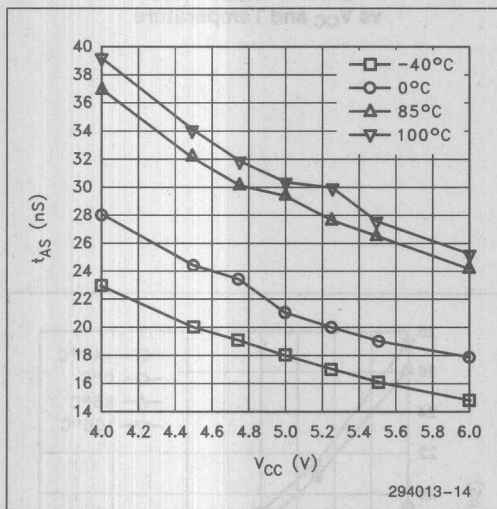


Figure 19. t_{AVWH} (t_{AS} WE#)
vs V_{CC} and Temperature

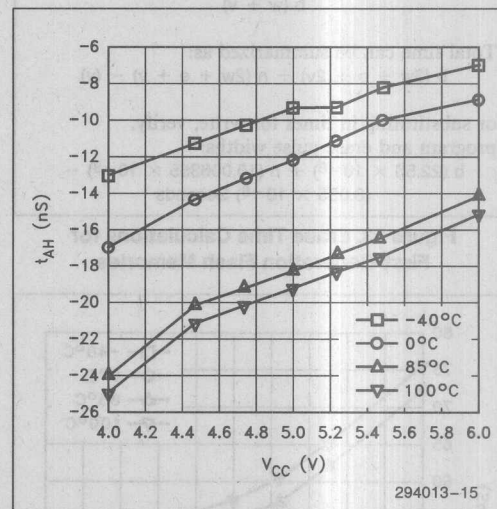


Figure 20. t_{WHAX} (t_{AH} WE#)
vs V_{CC} and Temperature

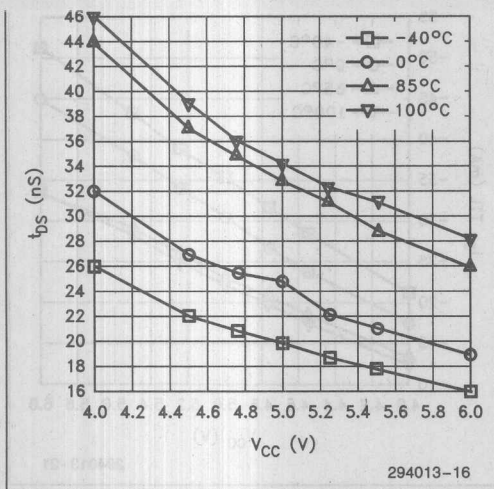


Figure 21. t_{DVEH} (t_{DS} CE #)
vs V_{CC} and Temperature

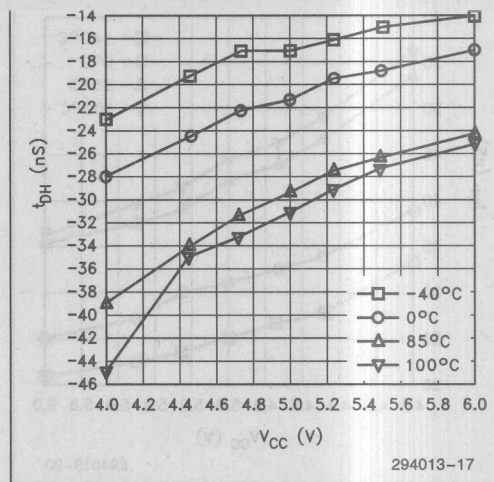


Figure 22. t_{EHDx} (t_{DH} CE #)
vs V_{CC} and Temperature

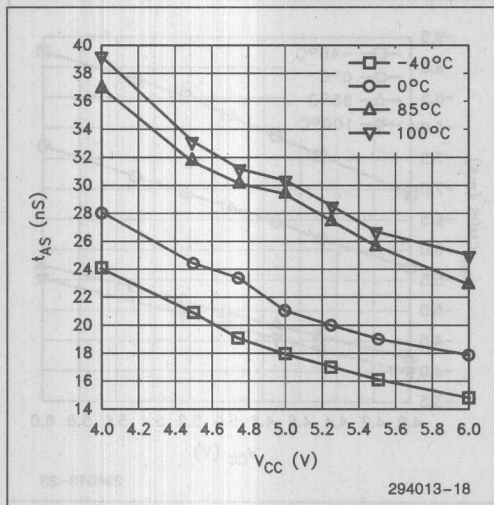


Figure 23. t_{AVEH} (t_{AS} CE #)
vs V_{CC} and Temperature

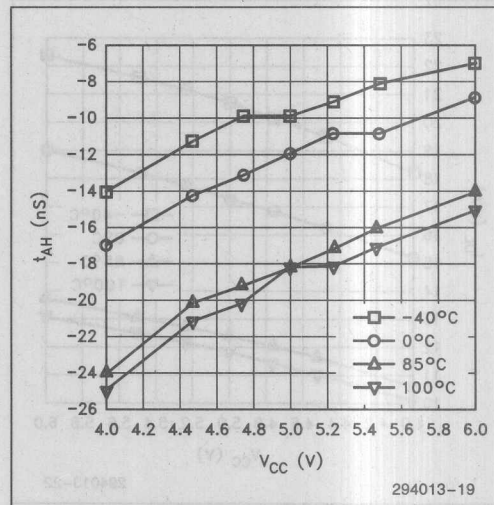


Figure 24. t_{EHAX} (t_{AH} CE #)
vs V_{CC} and Temperature

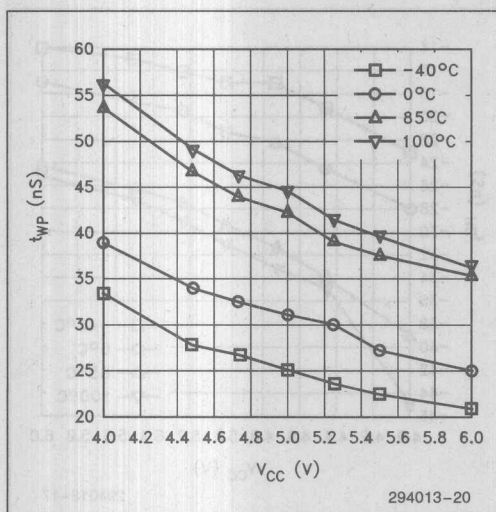


Figure 25. t_{WLWH} (t_{WP})
vs V_{CC} and Temperature

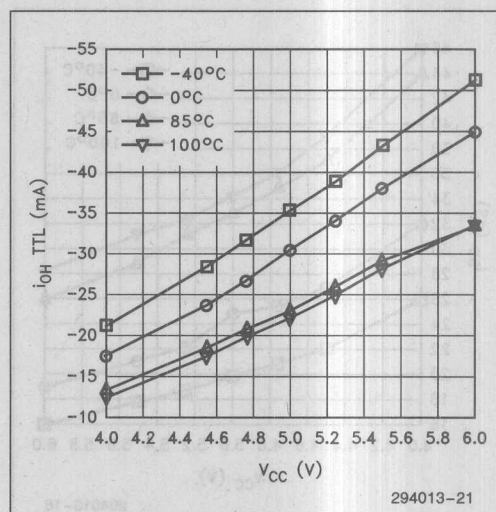


Figure 26. I_{OH} TTL
vs V_{CC} and Temperature

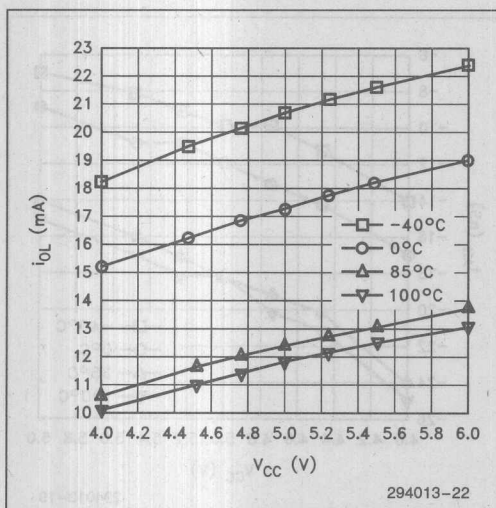


Figure 27. I_{OL} vs V_{CC} and Temperature
($V_{OL} = 0.45V$)

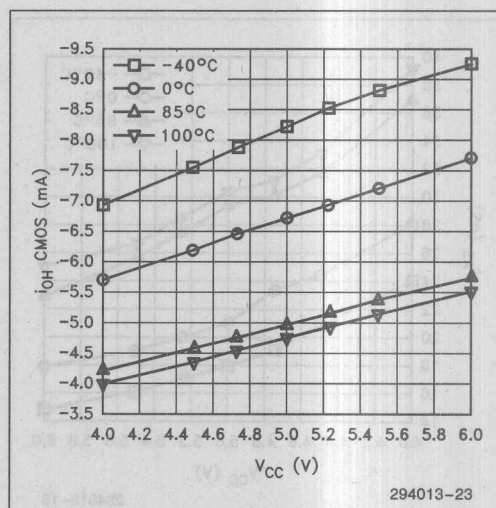


Figure 28. I_{OH} CMOS vs V_{CC} and Temperature

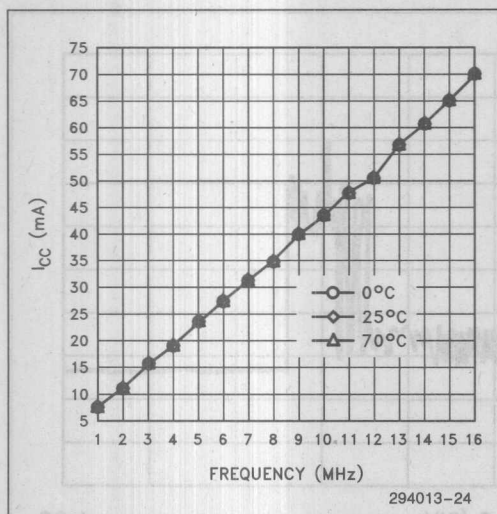


Figure 29. I_{CC} (RMS) vs Frequency
x16 Operation

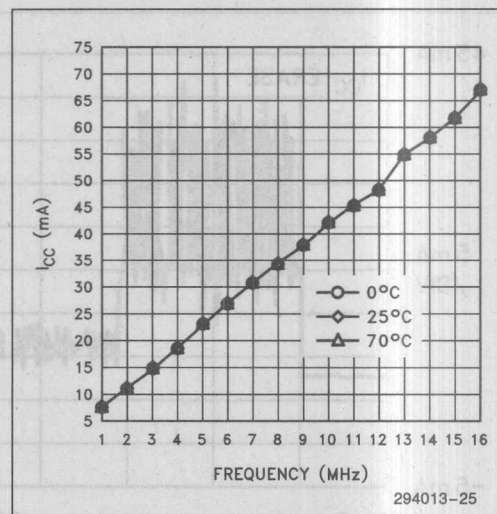


Figure 30. I_{CC} (RMS) vs Frequency
x8 Operation

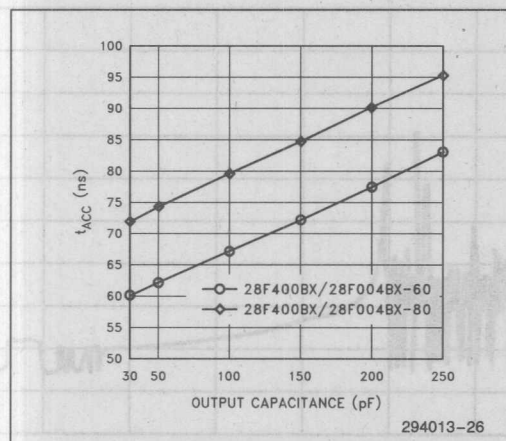
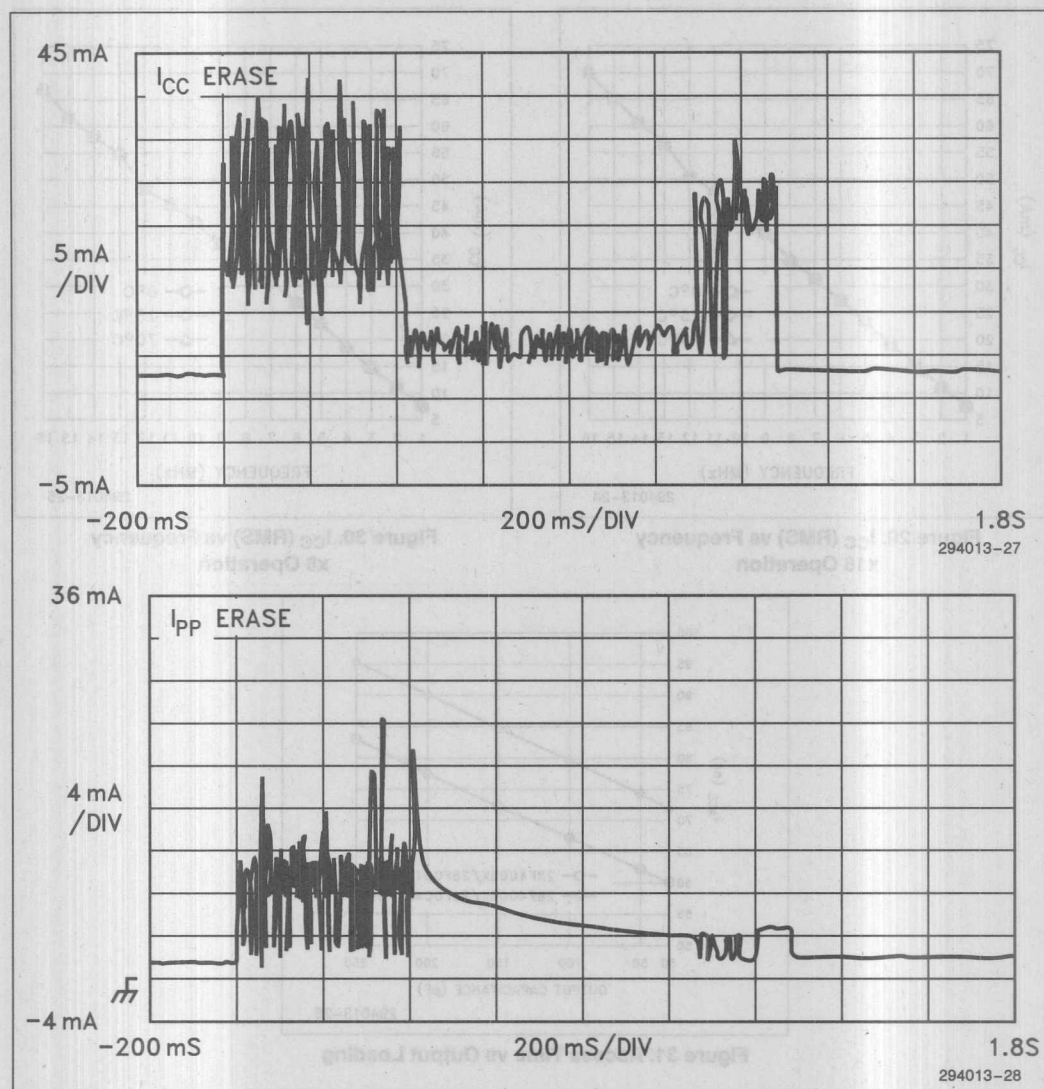


Figure 31. Access Time vs Output Loading

Figure 32. I_{CC} and I_{PP} during Block-Erase Operation

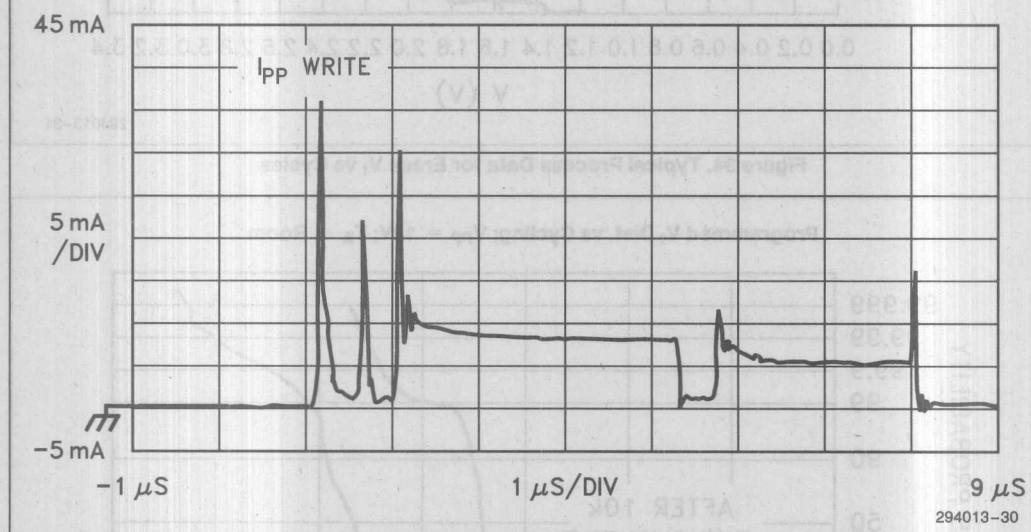
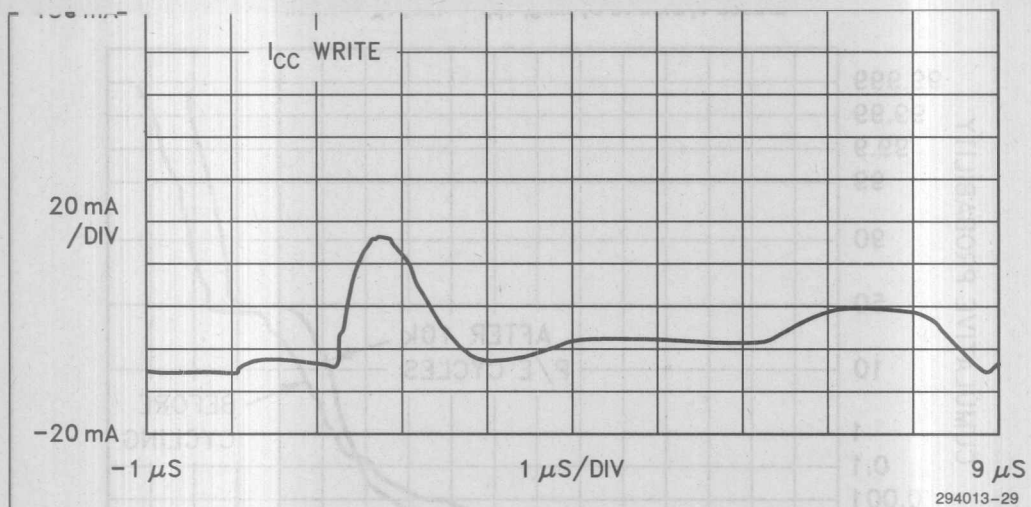
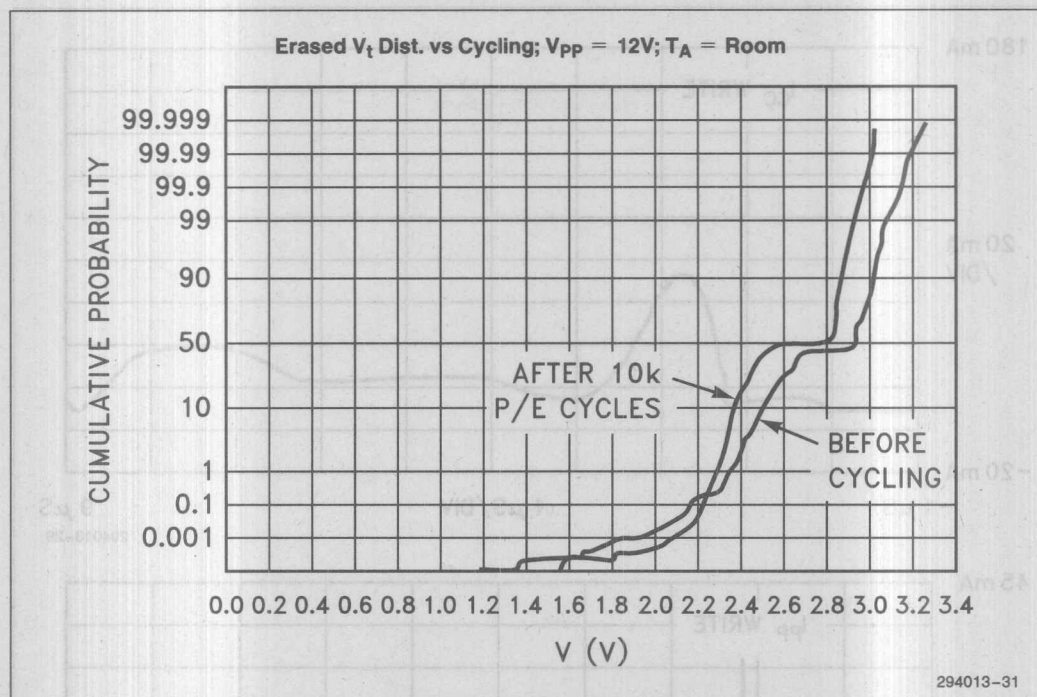
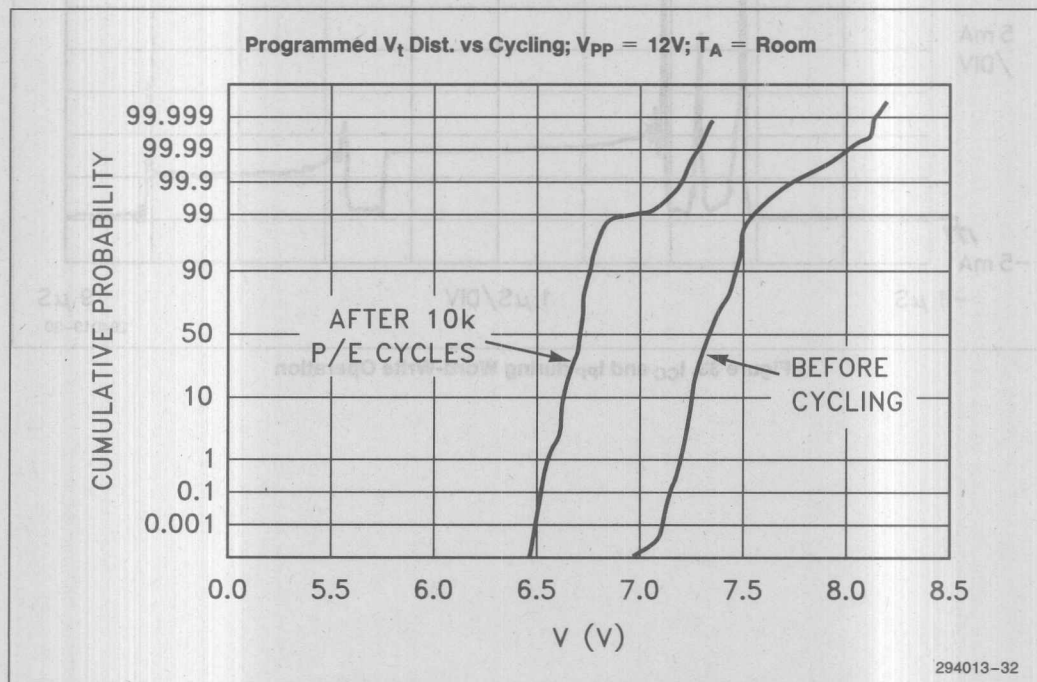
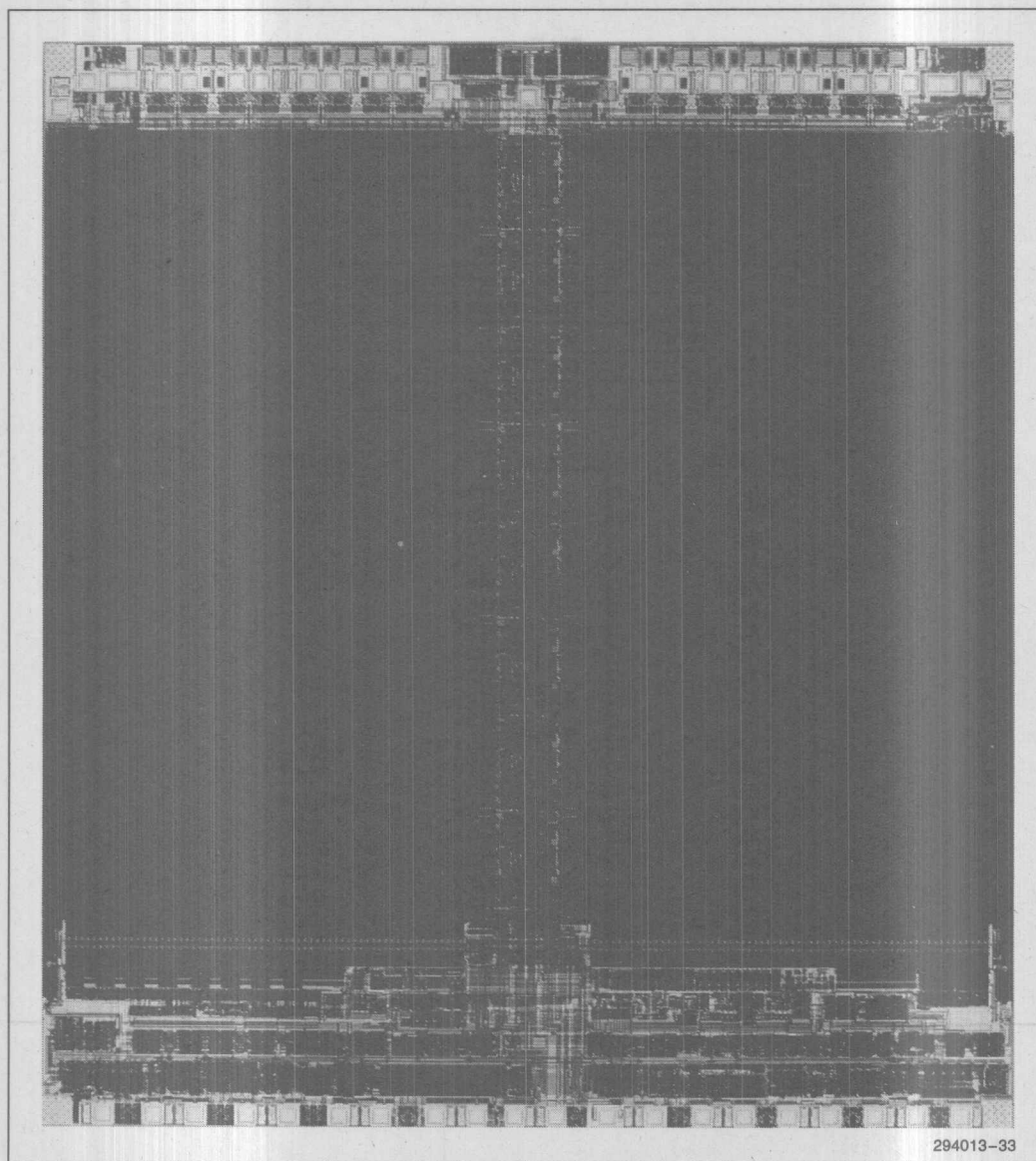


Figure 33. I_{CC} and I_{PP} during Word-Write Operation

Figure 34. Typical Process Data for Erase V_t vs CyclesFigure 35. Typical Process Data for Program V_t vs Cycles



294013-33

Figure 36. 28F400BX/28F004BX Die Photograph

A ₀ -A ₁₇	Address Inputs
DQ ₀ -DQ ₁₅	Data Inputs/Outputs
BYTE #	Byte Enable
CE #	Chip Enable
RP #	Power-Down/Reset
OE #	Output Enable
WE #	Write Enable
V _{PP}	Write/Erase Power Supply
V _{CC}	Device Power Supply
GND	Ground
DU	Don't Use
NC	No Internal Connection

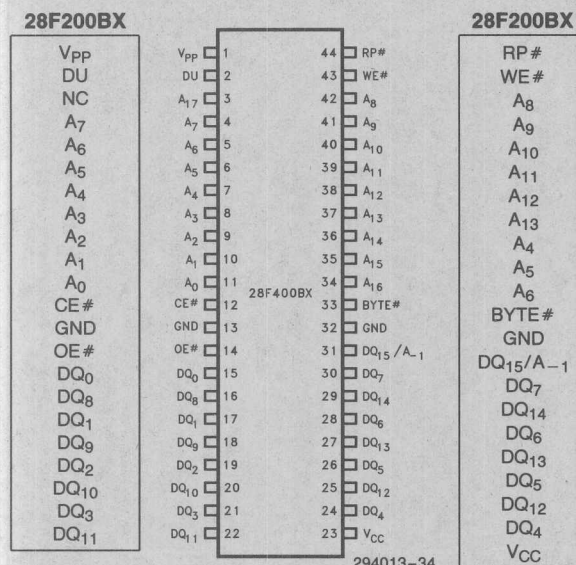
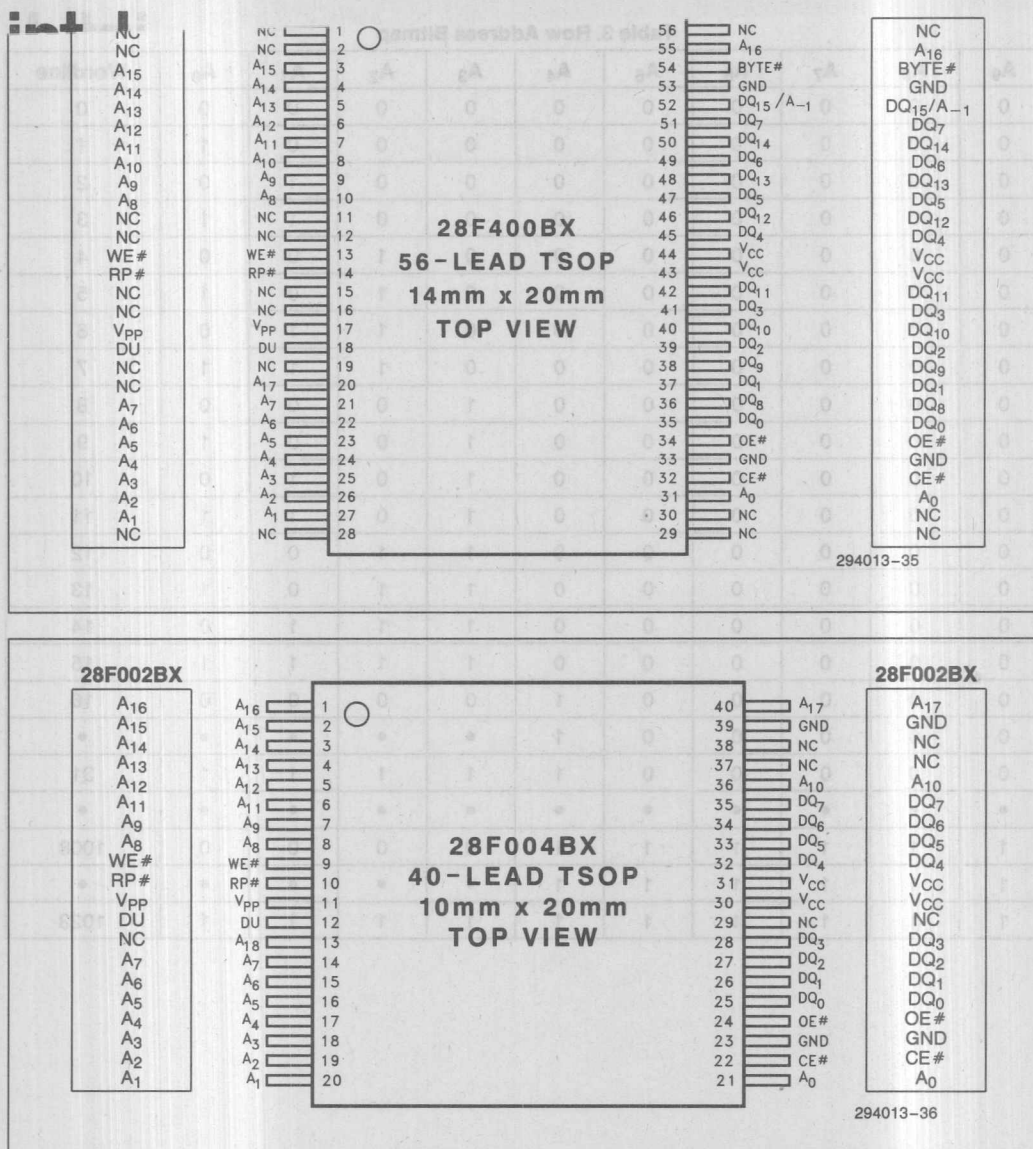


Figure 37. PSOP Lead Configuration



28F002BX

A₁₆

A₁₅

A₁₄

A₁₃

A₁₂

A₁₁

A₉

A₈

WE#

RP#

V_{PP}

DU

NC

A₇

A₆

A₅

A₄

A₃

A₂

A₁

A₁₆

A₁₅

A₁₄

A₁₃

A₁₂

A₁₁

A₉

A₈

WE#

RP#

V_{PP}

DU

A₁₈

A₇

A₆

A₅

A₄

A₃

A₂

A₁

28F004BX

40-LEAD TSOP

10mm x 20mm

TOP VIEW

40

39

38

37

36

35

34

33

32

31

30

29

28

27

26

25

24

23

22

21

A₁₇

GND

NC

NC

A₁₀

DQ₇

DQ₆

DQ₅

DQ₄

V_{CC}

V_{CC}

NC

DQ₃

DQ₂

DQ₁

DQ₀

OE#

GND

CE#

A₀

A₁₇

GND

NC

NC

A₁₀

DQ₇

DQ₆

DQ₅

DQ₄

V_{CC}

V_{CC}

NC

DQ₃

DQ₂

DQ₁

DQ₀

OE#

GND

CE#

A₀

294013-36

Figure 38. TSOP Lead Configuration

Addresses A₉–A₀ sequentially decode wordlines. Wordlines 0–1023 serve the upper and lower array planes.

Table 3. Row Address Bitmap

A ₉	A ₈	A ₇	A ₆	A ₅	A ₄	A ₃	A ₂	A ₁	A ₀	Wordline
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	1	1
0	0	0	0	0	0	0	0	1	0	2
0	0	0	0	0	0	0	0	1	1	3
0	0	0	0	0	0	0	1	0	0	4
0	0	0	0	0	0	0	1	0	1	5
0	0	0	0	0	0	0	1	1	0	6
0	0	0	0	0	0	0	1	1	1	7
0	0	0	0	0	0	1	0	0	0	8
0	0	0	0	0	0	1	0	0	1	9
0	0	0	0	0	0	1	0	1	0	10
0	0	0	0	0	0	1	0	1	1	11
0	0	0	0	0	0	1	1	0	0	12
0	0	0	0	0	0	1	1	0	1	13
0	0	0	0	0	0	1	1	1	0	14
0	0	0	0	0	0	1	1	1	1	15
0	0	0	0	0	1	0	0	0	0	16
0	0	0	0	0	1	•	•	•	•	•
0	0	0	0	0	1	1	1	1	1	31
•	•	•	•	•	•	•	•	•	•	•
1	1	1	1	1	1	0	0	0	0	1008
1	1	1	1	1	1	•	•	•	•	•
1	1	1	1	1	1	1	1	1	1	1023

Columns for each block are distributed throughout the array in each I/O. For the 28F400BX/200BX, each I/O contains 8 columns for the boot block, 4 columns each for parameter block 0 and parameter block 1, 48 columns for main block 0 and 64 columns for each of the remaining main blocks (main block 1 for the 28F200BX and main block 1-3 for the 28F400BX) for a total of 256 columns (28F400BX) or 128 columns (28F200BX) per output.

Table 4. 28F400BX/200BX Column Decoding

A ₁₇	A ₁₆	A ₁₅	A ₁₄	A ₁₃	A ₁₂	A ₁₁	A ₁₀	Block	Column
0	0	0	0	0	0	0	0	Boot	0
0	0	0	0	0	•	•	•	Boot	•
0	0	0	0	0	1	1	1	Boot	7
0	0	0	0	1	0	0	0	Parameter 0	8
0	0	0	0	1	0	•	•	Parameter 0	•
0	0	0	0	1	0	1	1	Parameter 0	11
0	0	0	0	1	1	0	0	Parameter 1	12
0	0	0	0	1	1	•	•	Parameter 1	•
0	0	0	0	1	1	1	1	Parameter 1	15
0	0	0	1	0	0	0	0	Main 0	16
0	0	•	•	•	•	•	•	Main 0	•
0	0	1	1	1	1	1	1	Main 0	63
0	1	0	0	0	0	0	0	Main 1	64
0	1	•	•	•	•	•	•	Main 1	•
0	1	1	1	1	1	1	1	Main 1	127
1	0	0	0	0	0	0	0	Main 2	128
1	0	•	•	•	•	•	•	Main 2	•
1	0	1	1	1	1	1	1	Main 2	191
1	1	0	0	0	0	0	0	Main 3	192
1	1	•	•	•	•	•	•	Main 3	•
1	1	1	1	1	1	1	1	Main 3	255

NOTE: 28F400BX = Complete Table, 28F200BX = Shaded Area Only

Columns for each block are distributed throughout the array in each I/O. For the 28F004BX/002BX, each I/O contains 16 columns for the boot block, 8 columns each for parameter block 0 and parameter block 1, 96 columns for main block 0 and 128 columns for each of the remaining main blocks (main block 1 for the 28F002BX) and main block 1-3 for the 28F004BX for a total of 512 columns (28F004BX) or 256 columns (28F002BX) per output.

Table 5. 28F004BX/002BX Column Decoding

A ₁₈	A ₁₇	A ₁₆	A ₁₅	A ₁₄	A ₁₃	A ₁₂	A ₁₁	A ₁₀	Block	Column
0	0	0	0	0	0	0	0	0	Boot	0
0	0	0	0	0	•	•	•	•	Boot	•
0	0	0	0	0	1	1	1	1	Boot	15
0	0	0	0	1	0	0	0	0	Parameter 0	16
0	0	0	0	1	0	•	•	•	Parameter 0	•
0	0	0	0	1	0	1	1	1	Parameter 0	23
0	0	0	0	1	1	0	0	0	Parameter 1	24
0	0	0	0	1	1	•	•	•	Parameter 1	•
0	0	0	0	1	1	1	1	1	Parameter 1	31
0	0	0	1	0	0	0	0	0	Main 0	32
0	0	•	•	•	•	•	•	•	Main 0	•
0	0	1	1	1	1	1	1	1	Main 0	127
0	1	0	0	0	0	0	0	0	Main 1	128
0	1	•	•	•	•	•	•	•	Main 1	•
0	1	1	1	1	1	1	1	1	Main 1	255
1	0	0	0	0	0	0	0	0	Main 2	256
1	0	•	•	•	•	•	•	•	Main 2	•
1	0	1	1	1	1	1	1	1	Main 2	383
1	1	0	0	0	0	0	0	0	Main 3	384
1	1	•	•	•	•	•	•	•	Main 3	•
1	1	1	1	1	1	1	1	1	Main 3	511

NOTE: 28F004BX = Complete Table, 28F002BX = Shaded Area Only



BOOT BLOCK FLASH: THE NEXT GENERATION

INTRODUCTION

Flash memory offers a whole new dimension to nonvolatile memory applications because of its high density and cost-effectiveness. Embedded systems, such as PC BIOS, hard disk drive controllers and laser printers, were among the first applications to utilize the easy update capability of Flash, allowing system differentiation and upgradeability. In 1992, Intel introduced a series of high-density Flash Memory Cards based on its 8-Mbit FlashFile™ component, products that are enabling truly portable computing capability with diskless, solid-state mass storage. Intel also added two flash memory products designed specifically to serve more traditional embedded and portable applications, offering both high performance and low power consumption options in an architecture specifically designed for these applications storage requirements.

MARKET GROWTH DATA

In the four years since Intel first introduced its 256-Kbit flash devices, the worldwide demand has quadrupled annually, with 1992 market demand forecasted to exceed \$270 million. Flash is expected to continue double-digit growth through the upcoming years while other memory technologies are relatively flat or increasing only slightly. By 1994, the flash market is expected to exceed \$1 billion (Figure 1). The primary reasons for this growth are well acknowledged by customers: flexibility, nonvolatility, low power and cost-effectiveness. As a result of this seemingly insatiable demand for flash, the memory density treadmill that allowed DRAMs and EPROMs to double every 18 months, has accelerated faster than ever before. Intel's introduction of an 8-Mbit component two years after its 2-Mbit product is evidence of the flash memory's rapid advancement.

Intel's first generation of flash devices, ranging in density from 256-Kbit through 2-Mbit densities, were rapidly adopted into code storage applications which formerly used EPROM or EEPROM. In 1991, Intel introduced the first 1-Mbit Boot Block flash memory in response to direct customer requests. As a result of continuing to improve and develop an innovative and broad product line, Intel held an 85% share of the \$130 million 1991 market.

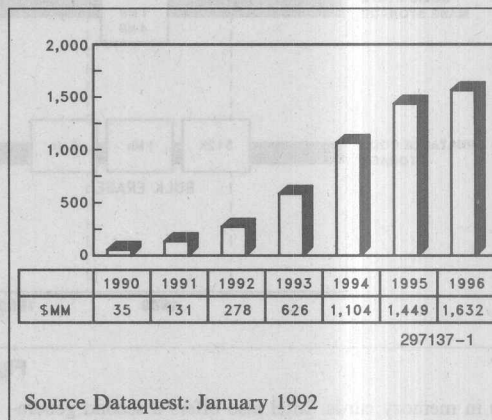


Figure 1

INTEL SERVING TWO PRIMARY APPLICATIONS

Today, Intel flash is serving two major application segments: **updatable code storage** and **solid-state mass storage** (Figure 2). Code and data storage comprise the updatable non-volatile memory applications that require high performance, high density and easy update capability. These applications are infrequently updated when compared to solid-state mass storage applications. In this case, erase/write performance is not as critical as integration and performance requirements. This application segment is served effectively with full chip-erase or boot block products.

The second major application segment is **solid-state mass storage** that requires very high density, automated programming and high performance erase/write capability at a very low cost per bit. Erasing and writing portions of the code or data is much more frequent in solid-state mass storage than in updatable firmware applications. In April of 1992, Intel introduced the 8-Mbit FlashFile component whose architecture is optimized for data file storage. Its symmetrically blocked architecture and automated write/erase features gave programming flexibility for a high-performance, solid-state memory system. The compactness of an 8-Mbit device in a TSOP package allows for high-density flash arrays to be included on a system motherboard as well

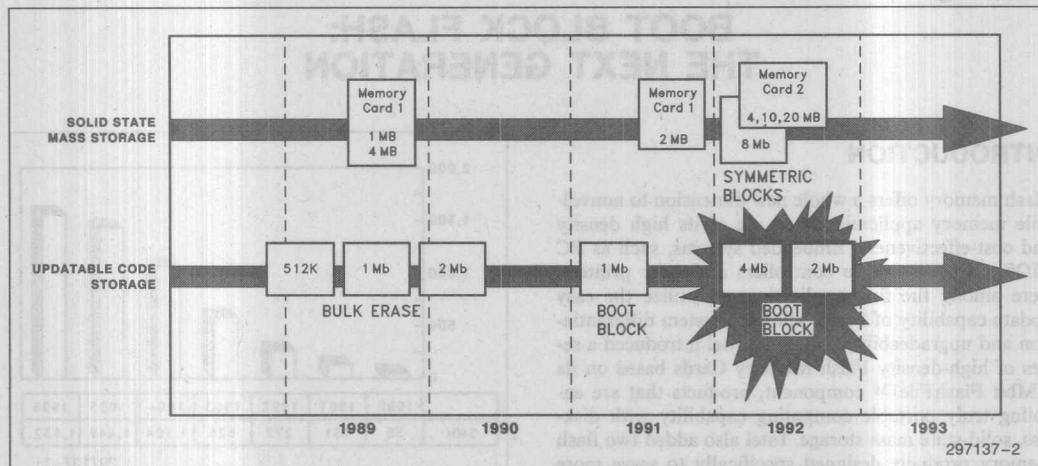


Figure 2

as in memory cards. Intel also offers a second generation of flash memory cards designed to serve the portable mass storage market that are based on the new 8-Mbit FlashFile component. Memory cards add the feature of removability and system upgradability in an industry-standard PCMCIA/ExCA™ format. While flash cards and FlashFile components serve solid-state mass storage applications directly, both can be used in updatable code storage applications as well.

By developing products to fit both of these application segments, Intel strives to serve the needs of a broader base of customer applications with the best nonvolatile memory solutions.

BOOT BLOCK UPDATE

The 28F001BX 1-Mbit Boot Block flash component introduced in June 1991 featured a sectored architecture that has been widely accepted in embedded code storage applications, particularly PC BIOS and cellular communications. Over 20 PC manufacturers use this device in their products, including Compaq, Dell and Zenith Data Systems. The Boot Block architecture gave the manufacturer added flexibility and the ability to differentiate. End users also benefit from the ability to upgrade BIOS software quickly and securely. It is possible for the manufacturer to upload BIOS updates to an electronic bulletin board where the end-user gets the

upgrade for the price of the call. The blocked architecture allows the OEM customer to store critical system code securely in the boot block of the device that can minimally bring up the system and download to other locations of the device to initialize the system. The hardware boot locking feature guarantees that even if the power is disrupted during a BIOS update, the system will be able to recover immediately.

The success of the 1-Mbit Boot Block device has resulted in over 150 designs with annual shipments to exceed 1.5 million units worldwide in 1992.

EXTENDING THE BOOT BLOCK PRODUCT LINE: 2- AND 4-MEGABIT FLASH DEVICES

Once the Boot Block architecture became established in the marketplace, customers quickly began to ask for more features and enhancements: speed, density, low power, surface-mount options and an industry-standard upgrade path. These requests were based on the need to expand features in portable computing and communication products.

The 2-Mbit 28F200BX and 4-Mbit 28F400BX are Intel's newest additions to the flash Boot Block product line. These products offer 60 ns performance, two surface-mount packages, and a Boot Block architecture

similar to the 1-Mbit Boot Block device: one lockable Boot Block, two parameter blocks, and the balance of the device is divided into main blocks. The Boot Block securely stores the basic boot code required to initialize the system that can be protected by the hardware-locking feature, ensuring basic system operating code protection. The two parameter blocks can be used for a variety of purposes: manufacturing product code, setup parameters and storing frequently updated data such as system diagnostics. The 28F200BX contains one 16 Kbyte Boot Block, two 8 Kbyte parameter blocks, one 96 Kbyte and one 128 Kbyte main block (Figure 3a). The 28F400BX contains all of the blocks of the 28F200BX plus two additional 128 Kbyte main blocks (Figure 3b). Top and bottom Boot Block versions are available for both densities. Both devices are in the x16/x8 user-selectable organization of the industry-standard, ROM-compatible pin-out in 44-lead PSOP surface mount package. This pinout and package allow an easy upgrade from 2-Mbit to 4-Mbit since only one address is added at the top of the device package. Forty-lead TSOP x8-only versions are also available for both of these devices, providing high density in the smallest form factor. A 56-lead TSOP x16/x8 version will be available for applications requiring x16 organization in a TSOP package.

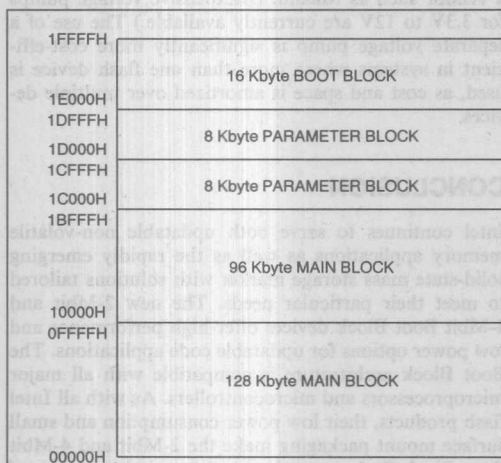


Figure 3a. 28F200BX-Top Boot

APPLICATION REQUIREMENTS—PCs

Notebook computer manufacturers today are competing to produce the lightest weight, slimmest and longest battery life products possible. Minimizing board size via reduced chip count is the first improvement the new flash devices provide. BIOS in portable systems has grown beyond the 1-Mbit density to accommodate more sophisticated power management and additional system features, such as PCMCIA card slots. For example, the i386™ SL architecture allows the expansion of BIOS beyond 128 Kbytes (1-Mbit) with internal registers for hardware paging. A new generation of BIOS support for low-power portable computers has been developed by SystemSoft and other vendors which supports the 28F200BX and the 28F400BX. In PC BIOS applications, the main blocks of the Boot Block devices are used for power management code, video drivers, and ROM-executable code, such as MS-DOS*. The arrangement of the blocks for PC BIOS applications based on Intel i386 and i486™ microprocessors is with the Boot Block on top. Other microprocessors and microcontrollers, such as the Intel 80960KX/SX and the Motorola 68000 series, use the bottom Boot Block version of the memory map. Many PC manufacturers have also moved to a x16 organization of their BIOS for

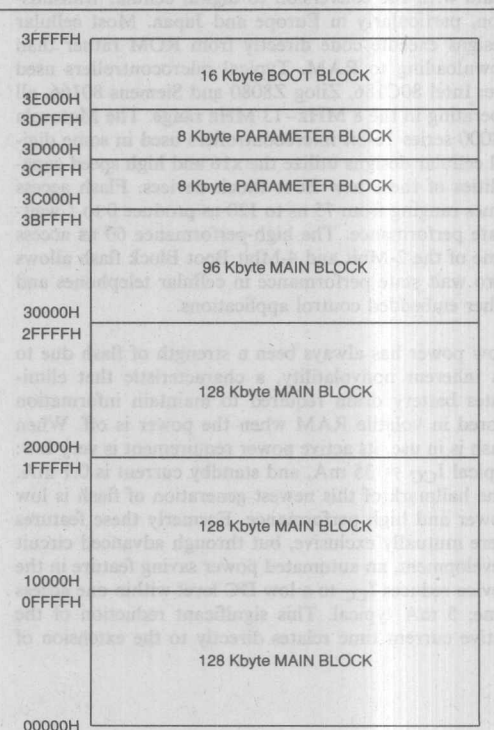


Figure 3b. 28F400BX-Top Boot

higher performance and are using 2-Mbit flash components in parallel to accomplish x16 performance. Other manufacturers have decided to forego x16 and use the 1-Mbit Boot Block and a 1-Mbit bulk-erase 28F010 flash memory to contain their power management code. The 28F200BX solves both of these problems in a compact, single-chip solution. The 28F400BX provides a higher density solution for high performance systems today, as well as a future upgrade for systems currently requiring only 2 megabits.

APPLICATION REQUIREMENTS—TELECOMMUNICATIONS/EMBEDDED

Similar to personal computers, cellular telephone applications use the hardware-lockable block to store basic initialization code to wake up the system and establish basic communication with the host base station. The small parameter blocks are ideal for frequently updated code such as the user's phone directory, re-dial and the activation code necessary to enable user-desired features. The main block contains the code for dynamically managing the cellular communications and executing the voice algorithms. The density requirement for the main code storage has increased significantly in recent years with the conversion to digital cellular transmission, particularly in Europe and Japan. Most cellular designs execute code directly from ROM rather than downloading to RAM. Typical microcontrollers used are: Intel 80C186, Zilog Z8080 and Siemens 80166, all operating in the 8 MHz–13 MHz range. The Motorola 68000 series 16-bit microcontrollers used in some digital cellular designs utilize the x16 and high speed capabilities of these new Boot Block devices. Flash access times ranging from 75 ns to 120 ns produce 0 to 1 wait-state performance. The high-performance 60 ns access time of the 2-Mbit and 4-Mbit Boot Block flash allows zero wait state performance in cellular telephones and other embedded control applications.

Low power has always been a strength of flash due to its inherent nonvolatility, a characteristic that eliminates battery drain required to maintain information stored in volatile RAM when the power is off. When flash is in use, its active power requirement is very low: typical $I_{CC} = 35$ mA, and standby current is 0.1 mA. The hallmark of this newest generation of flash is low power and high performance. Formerly these features were mutually exclusive, but through advanced circuit development, an automated power saving feature in the device reduces I_{CC} to a low DC level within one access time; 5 mA typical. This significant reduction of the active current time relates directly to the extension of

battery life in systems which continuously execute code directly from flash, such as cellular telephones and portable instrumentation.

For optimum system power conservation, future systems are being designed today to operate at 3.3V rather than 5V. Intel will also offer 3.3V versions of the 2-Mbit and 4-Mbit Boot Block devices with I_{CC} active = 10 mA, and t_{ACC} performance = 150 ns.

An alternative design approach of a "5V-only" (programming voltage = operating voltage = 5V) flash memory has been proposed to eliminate the need for a 12V programming supply. While this proposal would eliminate the need for a 12V supply component in a 5V-only environment, it would not provide a low power solution as every flash component would carry the die size cost and power overhead of on-chip voltage pumping. Intel's approach is to first bring the operating voltage to 3.3V, in line with what system designers requested as their first priority for the next generation of portable computers and cellular phones. The focus of lowering the read voltage is also congruent with the usage model for embedded applications of "read often, write few" operation. The 12V programming voltage requirement is more efficiently met with a voltage pump from a vendor such as Maxim. (Inexpensive voltage pumps for 3.3V to 12V are currently available.) The use of a separate voltage pump is significantly more cost-efficient in systems where more than one flash device is used, as cost and space is amortized over multiple devices.

CONCLUSION

Intel continues to serve both updatable non-volatile memory applications as well as the rapidly emerging solid-state mass storage market with solutions tailored to meet their particular needs. The new 2-Mbit and 4-Mbit Boot Block devices offer high performance and low power options for updatable code applications. The Boot Block architecture is compatible with all major microprocessors and microcontrollers. As with all Intel flash products, their low power consumption and small surface mount packaging make the 2-Mbit and 4-Mbit Boot Block flash memories ideal for a wide variety of handheld portable applications.

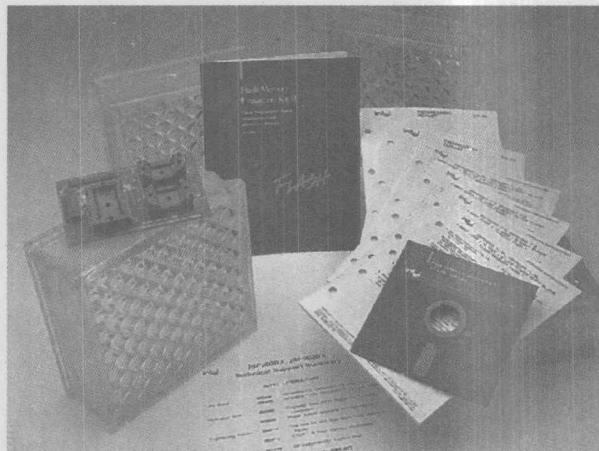
NOTE:

FlashFile, ExCA, i386 and i486 SL are trademarks of Intel Corporation.

*MS-DOS is a registered trademark of Microsoft Corporation.

Intel 2/4Mbit Boot Block Flash Memory Evaluation Module (D,FLASHEVAL5) Product Brief

- Kit Contents**
- 28F200BX/28F002BX
28F400BX/28F004BX
Adapter Board with:
 - 40-ld TSOP Socket
 - 56-ld TSOP Socket
 - 44-ld PSOP Socket
 - 5.25" Floppy Disk with iFlash2
Software (Version 2.4)
 - Technical Documentation
Describing Intel's 28F200BX/
28F002BX and 28F400BX/
28F004BX Flash Memory
devices
 - Flash Memory Evaluation Kit II
Installation Guide and User's
Manual with 28F200BX/
28F002BX/28F400BX/28F004BX
Adapter Board Installation
Instructions
 - Registration Card



Intel's 28F200BX/28F002BX/28F400BX/28F004BX Boot Block Flash Memory Evaluation Module provides system designers with a cost-effective prototyping tool for writing and erasing this flash memory device. This evaluation module is a hardware adapter board upgrade to the Intel Flash Memory Evaluation Kit II (D, FLASHEVAL2) which supports the 2 and 4 megabit boot block flash memories in 40-lead TSOP, 56-lead TSOP, and 44-lead PSOP packages.

Kit Description

The 28F200BX/28F002BX/28F400BX/28F004BX Evaluation Module, used with Intel's Flash Memory Evaluation Kit, provides the hardware, software and system interface necessary to evaluate and integrate Intel's 2 and 4Mbit Boot Block Flash Memories into your next design.

The module provides instructions to install the 28F200BX/28F002BX/28F400BX/28F004BX adapter board. Technical documentation includes 28F200BX/28F002BX/28F400BX/28F004BX datasheets, engineering reports and application notes. Together, they provide a complete description of the technology and important design considerations.

28F020

2048K (256K x 8) CMOS FLASH MEMORY

- Flash Electrical Chip-Erase
 - 2 Second Typical Chip-Erase
- Quick-Pulse Programming Algorithm
 - 10 μ s Typical Byte-Program
 - 4 Second Chip-Program
- 100,000 Erase/Program Cycles
- 12.0V \pm 5% V_{pp}
- High-Performance Read
 - 70 ns Maximum Access Time
- CMOS Low Power Consumption
 - 10 mA Typical Active Current
 - 50 μ A Typical Standby Current
 - 0 Watts Data Retention Power
- Integrated Program/Erase Stop Timer
- Command Register Architecture for Microprocessor/Microcontroller Compatible Write Interface
- Noise Immunity Features
 - \pm 10% V_{CC} Tolerance
 - Maximum Latch-Up Immunity through EPI Processing
- ETOX Nonvolatile Flash Technology
 - EPROM-Compatible Process Base
 - High-Volume Manufacturing Experience
- JEDEC-Standard Pinouts
 - 32-Pin Plastic Dip
 - 32-Lead PLCC
 - 32-Lead TSOP
- (See Packaging Spec., Order #231369)
- Extended Temperature Options

Intel's 28F020 CMOS flash memory offers the most cost-effective and reliable alternative for read/write random access nonvolatile memory. The 28F020 adds electrical chip-erase and reprogramming to familiar EPROM technology. Memory contents can be rewritten: in a test socket; in a PROM-programmer socket; on-board during subassembly test; in-system during final test; and in-system after-sale. The 28F020 increases memory flexibility, while contributing to time-and cost-savings.

The 28F020 is a 2048-kilobit nonvolatile memory organized as 262,144 bytes of 8 bits. Intel's 28F020 is offered in 32-pin plastic DIP, 32-lead PLCC, and 32-lead TSOP packages. Pin assignments conform to JEDEC standards for byte-wide EPROMs.

Extended erase and program cycling capability is designed into Intel's ETOX (EPROM Tunnel Oxide) process technology. Advanced oxide processing, an optimized tunneling structure, and lower electric field combine to extend reliable cycling beyond that of traditional EEPROMs. With the 12.0V V_{pp} supply, the 28F020 performs 100,000 erase and program cycles well within the time limits of the Quick-Pulse Programming and Quick-Erase algorithms.

Intel's 28F020 employs advanced CMOS circuitry for systems requiring high-performance access speeds, low power consumption, and immunity to noise. Its 70 nanosecond access time provides no-WAIT-state performance for a wide range of microprocessors and microcontrollers. Maximum standby current of 100 μ A translates into power savings when the device is deselected. Finally, the highest degree of latch-up protection is achieved through Intel's unique EPI processing. Prevention of latch-up is provided for stresses up to 100 mA on address and data pins, from -1 V to V_{CC} + 1V.

With Intel's ETOX process base, the 28F020 levers years of EPROM experience to yield the highest levels of quality, reliability, and cost-effectiveness.

ERASE/PROGRAM POWER SUPPLY (V _{PP})	12.0V
DEVICE POWER SUPPLY (V _{CC})	5.0V
GROUND	0V

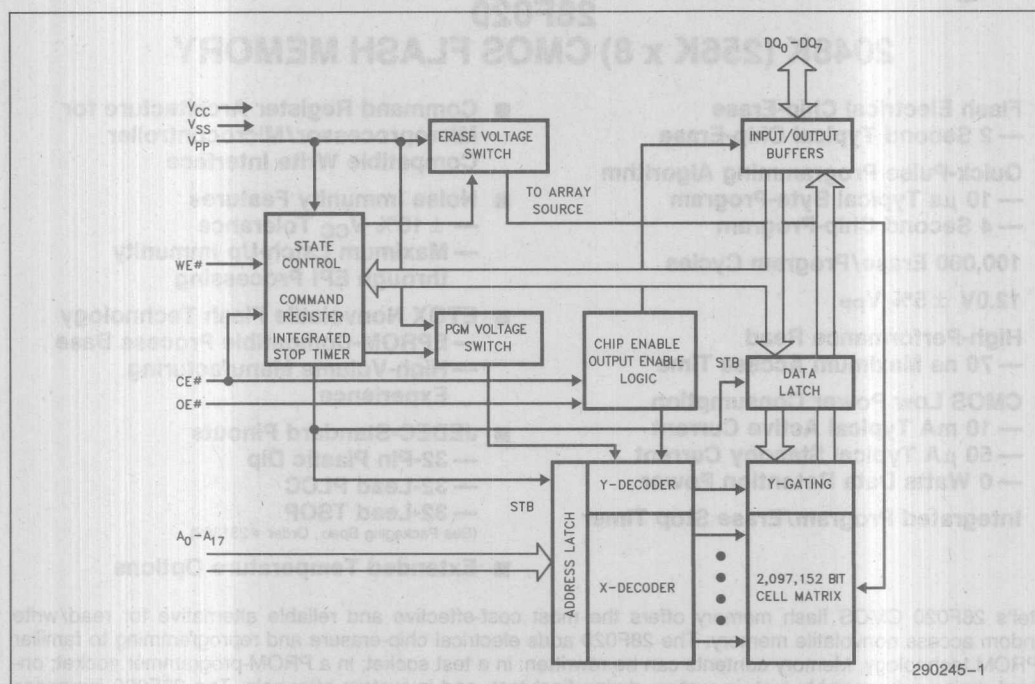


Figure 1. 28F020 Block Diagram

Table 1. Pin Description

Symbol	Type	Name and Function
A ₀ -A ₁₇	INPUT	ADDRESS INPUTS for memory addresses. Addresses are internally latched during a write cycle.
DQ ₀ -DQ ₇	INPUT/OUTPUT	DATA INPUT/OUTPUT: Inputs data during memory write cycles; outputs data during memory read cycles. The data pins are active high and float to tri-state OFF when the chip is deselected or the outputs are disabled. Data is internally latched during a write cycle.
CE #	INPUT	CHIP ENABLE: Activates the device's control logic, input buffers, decoders and sense amplifiers. CE # is active low; CE # high deselects the memory device and reduces power consumption to standby levels.
OE #	INPUT	OUTPUT ENABLE: Gates the devices output through the data buffers during a read cycle. OE # is active low.
WE #	INPUT	WRITE ENABLE: Controls writes to the control register and the array. Write enable is active low. Addresses are latched on the falling edge and data is latched on the rising edge of the WE # pulse. Note: With V _{PP} ≤ 6.5V, memory contents cannot be altered.
V _{PP}		ERASE/PROGRAM POWER SUPPLY for writing the command register, erasing the entire array, or programming bytes in the array.
V _{CC}		DEVICE POWER SUPPLY (5V ± 10%)
V _{SS}		GROUND

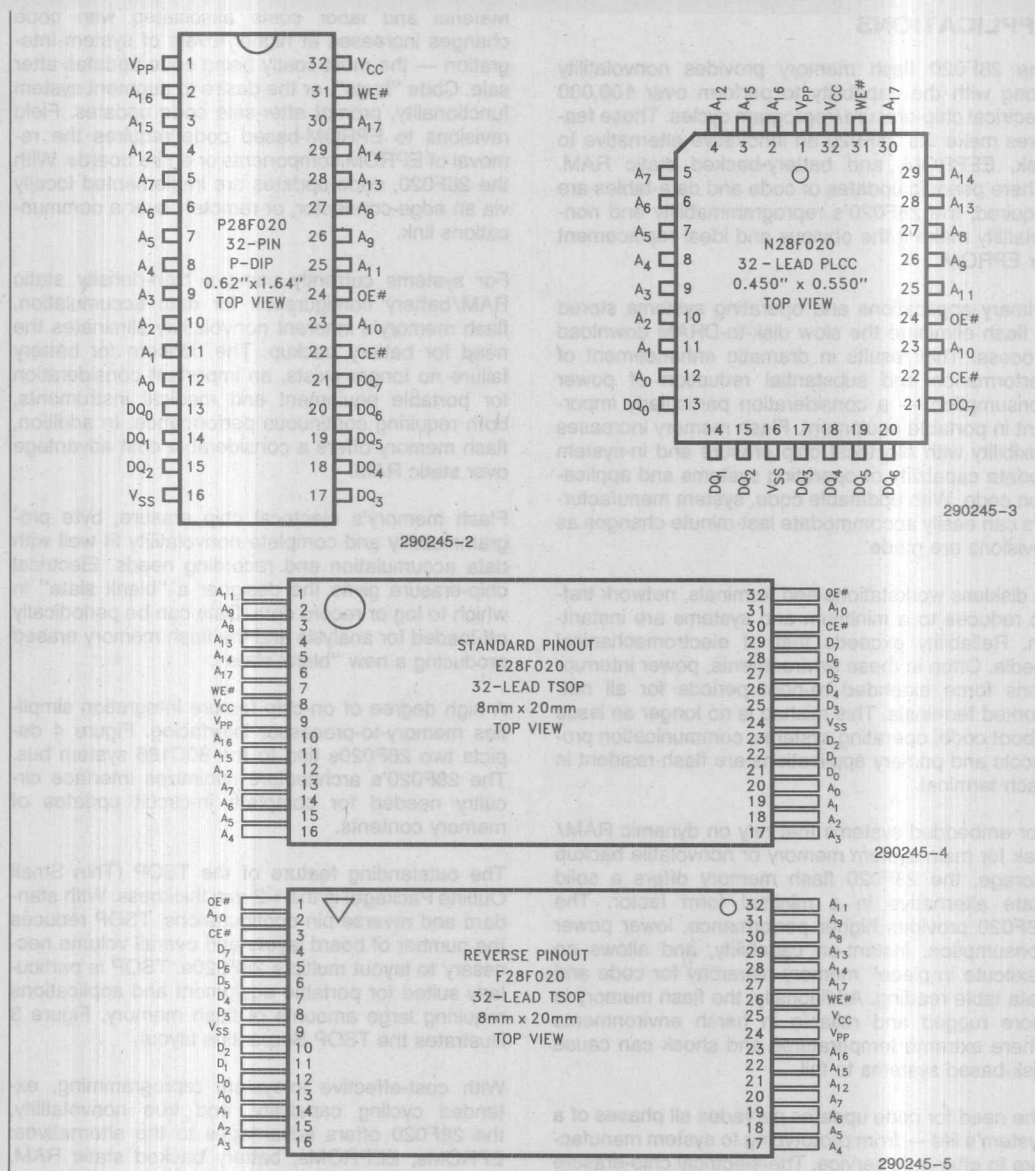


Figure 2. 28F020 Pin Configurations

APPLICATIONS

The 28F020 flash memory provides nonvolatility along with the capability to perform over 100,000 electrical chip-erase/reprogram cycles. These features make the 28F020 an innovative alternative to disk, EEPROM, and battery-backed static RAM. Where periodic updates of code and data-tables are required, the 28F020's reprogrammability and nonvolatility make it the obvious and ideal replacement for EPROM.

Primary applications and operating systems stored in flash eliminate the slow disk-to-DRAM download process. This results in dramatic enhancement of performance and substantial reduction of power consumption — a consideration particularly important in portable equipment. Flash memory increases flexibility with electrical chip erasure and in-system update capability of operating systems and application code. With updatable code, system manufacturers can easily accommodate last-minute changes as revisions are made.

In diskless workstations and terminals, network traffic reduces to a minimum and systems are instant-on. Reliability exceeds that of electromechanical media. Often in these environments, power interruptions force extended re-boot periods for all networked terminals. This mishap is no longer an issue if boot code, operating systems, communication protocols and primary applications are flash-resident in each terminal.

For embedded systems that rely on dynamic RAM/disk for main system memory or nonvolatile backup storage, the 28F020 flash memory offers a solid state alternative in a minimal form factor. The 28F020 provides higher performance, lower power consumption, instant-on capability, and allows an "execute in place" memory hierarchy for code and data table reading. Additionally, the flash memory is more rugged and reliable in harsh environments where extreme temperatures and shock can cause disk-based systems to fail.

The need for code updates pervades all phases of a system's life — from prototyping to system manufacture to after-sale service. The electrical chip-erase and reprogramming ability of the 28F020 allows in-circuit alterability; this eliminates unnecessary handling and less-reliable socketed connections, while adding greater test, manufacture, and update flexibility.

Material and labor costs associated with code changes increases at higher levels of system integration — the most costly being code updates after sale. Code "bugs", or the desire to augment system functionality, prompt after-sale code updates. Field revisions to EPROM-based code requires the removal of EPROM components or entire boards. With the 28F020, code updates are implemented locally via an edge-connector, or remotely over a communications link.

For systems currently using a high-density static RAM/battery configuration for data accumulation, flash memory's inherent nonvolatility eliminates the need for battery backup. The concern for battery failure no longer exists, an important consideration for portable equipment and medical instruments, both requiring continuous performance. In addition, flash memory offers a considerable cost advantage over static RAM.

Flash memory's electrical chip erasure, byte programmability and complete nonvolatility fit well with data accumulation and recording needs. Electrical chip-erase gives the designer a "blank slate" in which to log or record data. Data can be periodically off-loaded for analysis and the flash memory erased producing a new "blank slate".

A high degree of on-chip feature integration simplifies memory-to-processor interfacing. Figure 4 depicts two 28F020s tied to the 80C186 system bus. The 28F020's architecture minimizes interface circuitry needed for complete in-circuit updates of memory contents.

The outstanding feature of the TSOP (Thin Small Outline Package) is the 1.2 mm thickness. With standard and reverse pin configurations, TSOP reduces the number of board layers and overall volume necessary to layout multiple 28F020s. TSOP is particularly suited for portable equipment and applications requiring large amounts of flash memory. Figure 3 illustrates the TSOP Serpentine layout.

With cost-effective in-system reprogramming, extended cycling capability, and true nonvolatility, the 28F020 offers advantages to the alternatives: EPROMs, EEPROMs, battery backed static RAM, or disk. EPROM-compatible read specifications, straight-forward interfacing, and in-circuit alterability offers designers unlimited flexibility to meet the high standards of today's designs.

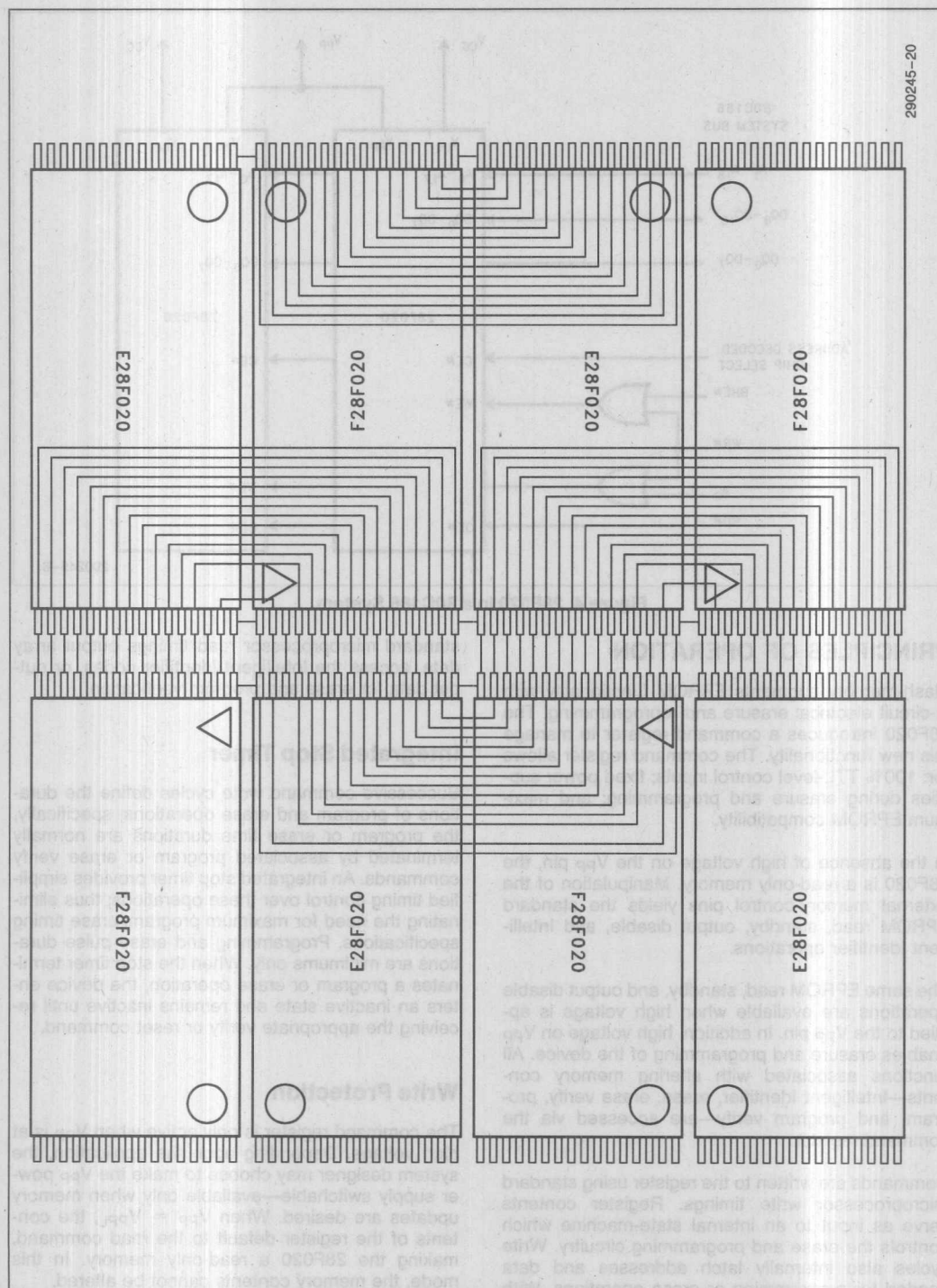


Figure 3. TSOP Serpentine Layout

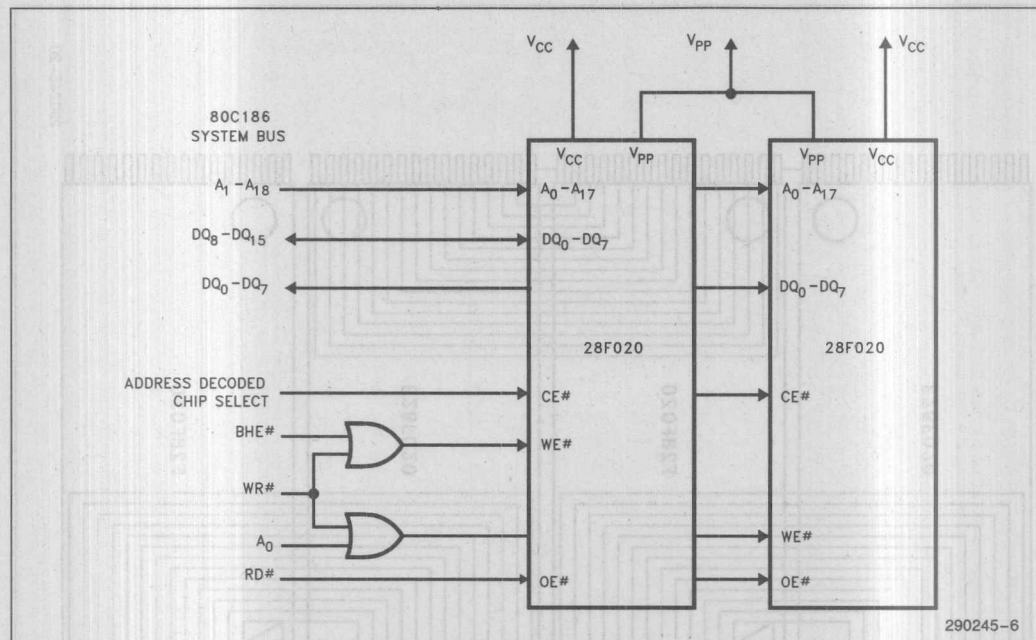


Figure 4. 28F020 in a 80C186 System

PRINCIPLES OF OPERATION

Flash-memory augments EPROM functionality with in-circuit electrical erasure and reprogramming. The 28F020 introduces a command register to manage this new functionality. The command register allows for: 100% TTL-level control inputs; fixed power supplies during erasure and programming; and maximum EPROM compatibility.

In the absence of high voltage on the V_{PP} pin, the 28F020 is a read-only memory. Manipulation of the external memory-control pins yields the standard EPROM read, standby, output disable, and Intelligent Identifier operations.

The same EPROM read, standby, and output disable operations are available when high voltage is applied to the V_{PP} pin. In addition, high voltage on V_{PP} enables erasure and programming of the device. All functions associated with altering memory contents—Intelligent Identifier, erase, erase verify, program, and program verify—are accessed via the command register.

Commands are written to the register using standard microprocessor write timings. Register contents serve as input to an internal state-machine which controls the erase and programming circuitry. Write cycles also internally latch addresses and data needed for programming or erase operations. With the appropriate command written to the register,

standard microprocessor read timings output array data, access the Intelligent Identifier codes, or output data for erase and program verification.

Integrated Stop Timer

Successive command write cycles define the durations of program and erase operations; specifically, the program or erase time durations are normally terminated by associated program or erase verify commands. An integrated stop timer provides simplified timing control over these operations; thus eliminating the need for maximum program/erase timing specifications. Programming and erase pulse durations are minimums only. When the stop timer terminates a program or erase operation, the device enters an inactive state and remains inactive until receiving the appropriate verify or reset command.

Write Protection

The command register is only active when V_{PP} is at high voltage. Depending upon the application, the system designer may choose to make the V_{PP} power supply switchable—available only when memory updates are desired. When $V_{PP} = V_{PPL}$, the contents of the register default to the read command, making the 28F020 a read-only memory. In this mode, the memory contents cannot be altered.

Table 2. 28F020 Bus Operations

		Pins	V _{PP} (1)	A ₀	A ₉	CE #	OE #	WE #	DQ ₀ –DQ ₇
Operation									
READ-ONLY	Read		V _{PPL}	A ₀	A ₉	V _{IL}	V _{IL}	V _{IH}	Data Out
	Output Disable		V _{PPL}	X	X	V _{IL}	V _{IH}	V _{IH}	Tri-State
	Standby		V _{PPL}	X	X	V _{IH}	X	X	Tri-State
	Intelligent Identifier (Mfr)(2)		V _{PPL}	V _{IL}	V _{ID} (3)	V _{IL}	V _{IL}	V _{IH}	Data = 89H
	Intelligent Identifier (Device)(2)		V _{PPL}	V _{IH}	V _{ID} (3)	V _{IL}	V _{IL}	V _{IH}	Data = BDH
READ/WRITE	Read		V _{PPH}	A ₀	A ₉	V _{IL}	V _{IL}	V _{IH}	Data Out(4)
	Output Disable		V _{PPH}	X	X	V _{IL}	V _{IH}	V _{IH}	Tri-State
	Standby(5)		V _{PPH}	X	X	V _{IH}	X	X	Tri-State
	Write		V _{PPH}	A ₀	A ₉	V _{IL}	V _{IH}	V _{IL}	Data In(6)

NOTES:

1. Refer to DC Characteristics. When V_{PP} = V_{PPL} memory contents can be read but not written or erased.
2. Manufacturer and device codes may also be accessed via a command register write sequence. Refer to Table 3. All other addresses low.
3. V_{ID} is the Intelligent Identifier high voltage. Refer to DC Characteristics.
4. Read operations with V_{PP} = V_{PPH} may access array data or the Intelligent Identifier codes.
5. With V_{PP} at high voltage, the standby current equals I_{CC} + I_{PP} (standby).
6. Refer to Table 3 for valid Data-In during a write operation.
7. X can be V_{IL} or V_{IH}.

Or, the system designer may choose to “hardwire” V_{PP}, making the high voltage supply constantly available. In this case, all Command Register functions are inhibited whenever V_{CC} is below the write lockout voltage V_{LKO}. (See Power Up/Down Protection.) The 28F020 is designed to accommodate either design practice, and to encourage optimization of the processor-memory interface.

The two step program/erase write sequence to the Command Register provides additional software write protection.

BUS OPERATIONS

Read

The 28F020 has two control functions, both of which must be logically active, to obtain data at the outputs. Chip-Enable (CE#) is the power control and should be used for device selection. Output-Enable (OE#) is the output control and should be used to gate data from the output pins, independent of device selection. Refer to AC read timing waveforms.

When V_{PP} is high (V_{PPH}), the read operation can be used to access array data, to output the Intelligent Identifier codes, and to access data for program/erase verification. When V_{PP} is low (V_{PPL}), the read operation can **only** access the array data.

Output Disable

With Output-Enable at a logic-high level (V_{IH}), output from the device is disabled. Output pins are placed in a high-impedance state.

Standby

With Chip-Enable at a logic-high level, the standby operation disables most of the 28F020's circuitry and substantially reduces device power consumption. The outputs are placed in a high-impedance state, independent of the Output-Enable signal. If the 28F020 is deselected during erasure, programming, or program/erase verification, the device draws active current until the operation is terminated.

Intelligent Identifier Operation

The Intelligent Identifier operation outputs the manufacturer code (89H) and device code (BDH). Programming equipment automatically matches the device with its proper erase and programming algorithms.

With Chip-Enable and Output-Enable at a logic low level, raising A9 to high voltage V_{ID} (see DC Characteristics) activates the operation. Data read from locations 0000H and 0001H represent the manufacturer's code and the device code, respectively.

The manufacturer- and device-codes can also be read via the command register, for instances where the 28F020 is erased and reprogrammed in the target system. Following a write of 90H to the command register, a read from address location 0000H outputs the manufacturer code (89H). A read from address 0001H outputs the device code (BDH).

Write

Device erasure and programming are accomplished via the command register, when high voltage is applied to the V_{pp} pin. The contents of the register serve as input to the internal state-machine. The state-machine outputs dictate the function of the device.

The command register itself does not occupy an addressable memory location. The register is a latch used to store the command, along with address and data information needed to execute the command.

The command register is written by bringing Write-Enable to a logic-low level (V_{IL}), while Chip-Enable is low. Addresses are latched on the falling edge of Write-Enable, while data is latched on the rising edge of the Write-Enable pulse. Standard microprocessor write timings are used.

Refer to AC Write Characteristics and the Erase/Programming Waveforms for specific timing parameters.

COMMAND DEFINITIONS

When low voltage is applied to the V_{pp} pin, the contents of the command register default to 00H, enabling read-only operations.

Placing high voltage on the V_{pp} pin enables read/write operations. Device operations are selected by writing specific data patterns into the command register. Table 3 defines these 28F020 register commands.

Table 3. Command Definitions

Command	Bus Cycles Req'd	First Bus Cycle			Second Bus Cycle		
		Operation(1)	Address(2)	Data(3)	Operation(1)	Address(2)	Data(3)
Read Memory	1	Write	X	00H			
Read Intelligent Identifier Codes(4)	3	Write	X	90H	Read	(4)	(4)
Set-up Erase/Erase(5)	2	Write	X	20H	Write	X	20H
Erase Verify(5)	2	Write	EA	A0H	Read	X	EVD
Set-up Program/Program(6)	2	Write	X	40H	Write	PA	PD
Program Verify(6)	2	Write	X	C0H	Read	X	PVD
Reset(7)	2	Write	X	FFH	Write	X	FFH

NOTES:

- Bus operations are defined in Table 2.
- IA = Identifier address: 00H for manufacturer code, 01H for device code.
EA = Address of memory location to be read during erase verify.
PA = Address of memory location to be programmed.
Addresses are latched on the falling edge of the Write-Enable pulse.
- ID = Data read from location IA during device identification (Mfr = 89H, Device = BDH).
EVD = Data read from location EA during erase verify.
PD = Data to be programmed at location PA. Data is latched on the rising edge of Write-Enable.
PVD = Data read from location PA during program verify. PA is latched on the Program command.
- Following the Read Intelligent ID command, two read operations access manufacturer and device codes.
- Figure 6 illustrates the Quick-Erase Algorithm.
- Figure 5 illustrates the Quick-Pulse Programming Algorithm.
- The second bus cycle must be followed by the desired command register write.

Read Command

While V_{PP} is high, for erasure and programming, memory contents can be accessed via the read command. The read operation is initiated by writing 00H into the command register. Microprocessor read cycles retrieve array data. The device remains enabled for reads until the command register contents are altered.

The default contents of the register upon V_{PP} power-up is 00H. This default value ensures that no spurious alteration of memory contents occurs during the V_{PP} power transition. Where the V_{PP} supply is hard-wired to the 28F020, the device powers-up and remains enabled for reads until the command-register contents are changed. Refer to the AC Read Characteristics and Waveforms for specific timing parameters.

Intelligent Identifier Command

Flash-memories are intended for use in applications where the local CPU alters memory contents. As such, manufacturer- and device-codes must be accessible while the device resides in the target system. PROM programmers typically access signature codes by raising A9 to a high voltage. However, multiplexing high voltage onto address lines is not a desired system-design practice.

The 28F020 contains an Intelligent Identifier operation to supplement traditional PROM-programming methodology. The operation is initiated by writing 90H into the command register. Following the command write, a read cycle from address 0000H retrieves the manufacturer code of 89H. A read cycle from address 0001H returns the device code of BDH. To terminate the operation, it is necessary to write another valid command into the register.

Set-up Erase/Erase Commands

Set-up Erase is a command-only operation that stages the device for electrical erasure of all bytes in the array. The set-up erase operation is performed by writing 20H to the command register.

To commence chip-erasure, the erase command (20H) must again be written to the register. The erase operation begins with the rising edge of the Write-Enable pulse and terminates with the rising edge of the next Write-Enable pulse (i.e., Erase-Verify Command).

This two-step sequence of set-up followed by execution ensures that memory contents are not accidentally erased. Also, chip-erasure can only occur when high voltage is applied to the V_{PP} pin. In the absence

of this high voltage, memory contents are protected against erasure. Refer to AC Erase Characteristics and Waveforms for specific timing parameters.

Erase-Verify Command

The erase command erases all bytes of the array in parallel. After each erase operation, all bytes must be verified. The erase verify operation is initiated by writing A0H into the command register. The address for the byte to be verified must be supplied as it is latched on the falling edge of the Write-Enable pulse. The register write terminates the erase operation with the rising edge of its Write-Enable pulse.

The 28F020 applies an internally-generated margin voltage to the addressed byte. Reading FFH from the addressed byte indicates that all bits in the byte are erased.

The erase-verify command must be written to the command register prior to each byte verification to latch its address. The process continues for each byte in the array until a byte does not return FFH data, or the last address is accessed.

In the case where the data read is not FFH, another erase operation is performed. (Refer to Set-up Erase/Erase). Verification then resumes from the address of the last-verified byte. Once all bytes in the array have been verified, the erase step is complete. The device can be programmed. At this point, the verify operation is terminated by writing a valid command (e.g. Program Set-up) to the command register. Figure 6, the Quick-Erase algorithm, illustrates how commands and bus operations are combined to perform electrical erasure of the 28F020. Refer to AC Erase Characteristics and Waveforms for specific timing parameters.

Set-up Program/Program Commands

Set-up program is a command-only operation that stages the device for byte programming. Writing 40H into the command register performs the set-up operation.

Once the program set-up operation is performed, the next Write-Enable pulse causes a transition to an active programming operation. Addresses are internally latched on the falling edge of the Write-Enable pulse. Data is internally latched on the rising edge of the Write-Enable pulse. The rising edge of Write-Enable also begins the programming operation. The programming operation terminates with the next rising edge of Write-Enable, used to write the program-verify command. Refer to AC Programming Characteristics and Waveforms for specific timing parameters.

Program-Verify Command

The 28F020 is programmed on a byte-by-byte basis. Byte programming may occur sequentially or at random. Following each programming operation, the byte just programmed must be verified.

The program-verify operation is initiated by writing COH into the command register. The register write terminates the programming operation with the rising edge of its Write-Enable pulse. The program-verify operation stages the device for verification of the byte last programmed. No new address information is latched.

The 28F020 applies an internally-generated margin voltage to the byte. A microprocessor read cycle outputs the data. A successful comparison between the programmed byte and true data means that the byte is successfully programmed. Programming then proceeds to the next desired byte location. Figure 5, the 28F020 Quick-Pulse Programming algorithm, illustrates how commands are combined with bus operations to perform byte programming. Refer to AC Programming Characteristics and Waveforms for specific timing parameters.

Reset Command

A reset command is provided as a means to safely abort the erase- or program-command sequences. Following either set-up command (erase or program) with two consecutive writes of FFH will safely abort the operation. Memory contents will not be altered. A valid command must then be written to place the device in the desired state.

EXTENDED ERASE/PROGRAM CYCLING

EEPROM cycling failures have always concerned users. The high electrical field required by thin oxide EEPROMs for tunneling can literally tear apart the oxide at defect regions. To combat this, some suppliers have implemented redundancy schemes, reducing cycling failures to insignificant levels. However, redundancy requires that cell size be doubled—an expensive solution.

Intel has designed extended cycling capability into its ETOX flash memory technology. Resulting improvements in cycling reliability come without increasing memory cell size or complexity. First, an advanced tunnel oxide increases the charge carrying ability ten-fold. Second, the oxide area per cell subjected to the tunneling electric field is one-tenth that of common EEPROMs, minimizing the probability of oxide defects in the region. Finally, the peak electric field during erasure is approximately 2 MV/cm lower than EEPROM. The lower electric

field greatly reduces oxide stress and the probability of failure—increasing time to wearout by a factor of 100,000,000.

The 28F020 is capable of 100,000 program/erase cycles. The device is programmed and erased using Intel's Quick-Pulse Programming and Quick-Erase algorithms. Intel's algorithmic approach uses a series of operations (pulses), along with byte verification, to completely and reliably erase and program the device.

For further information, see Reliability Report RR-60.

QUICK-PULSE PROGRAMMING ALGORITHM

The Quick-Pulse Programming algorithm uses programming operations of 10 μ s duration. Each operation is followed by a byte verification to determine when the addressed byte has been successfully programmed. The algorithm allows for up to 25 programming operations per byte, although most bytes verify on the first or second operation. The entire sequence of programming and byte verification is performed with V_{pp} at high voltage. Figure 5 illustrates the Quick-Pulse Programming algorithm.

QUICK-ERASE ALGORITHM

Intel's Quick-Erase algorithm yields fast and reliable electrical erasure of memory contents. The algorithm employs a closed-loop flow, similar to the Quick-Pulse Programming algorithm, to simultaneously remove charge from all bits in the array.

Erase begins with a read of memory contents. The 28F020 is erased when shipped from the factory. Reading FFH data from the device would immediately be followed by device programming.

For devices being erased and reprogrammed, uniform and reliable erasure is ensured by first programming all bits in the device to their charged state (Data = 00H). This is accomplished, using the Quick-Pulse Programming algorithm, in approximately four seconds.

Erase execution then continues with an initial erase operation. Erase verification (data = FFH) begins at address 0000H and continues through the array to the last address, or until data other than FFH is encountered. With each erase operation, an increasing number of bytes verify to the erased state. Erase efficiency may be improved by storing the address of the last byte verified in a register. Following the next erase operation, verification starts at that stored address location. Erasure typically occurs in two seconds. Figure 6 illustrates the Quick-Erase algorithm.

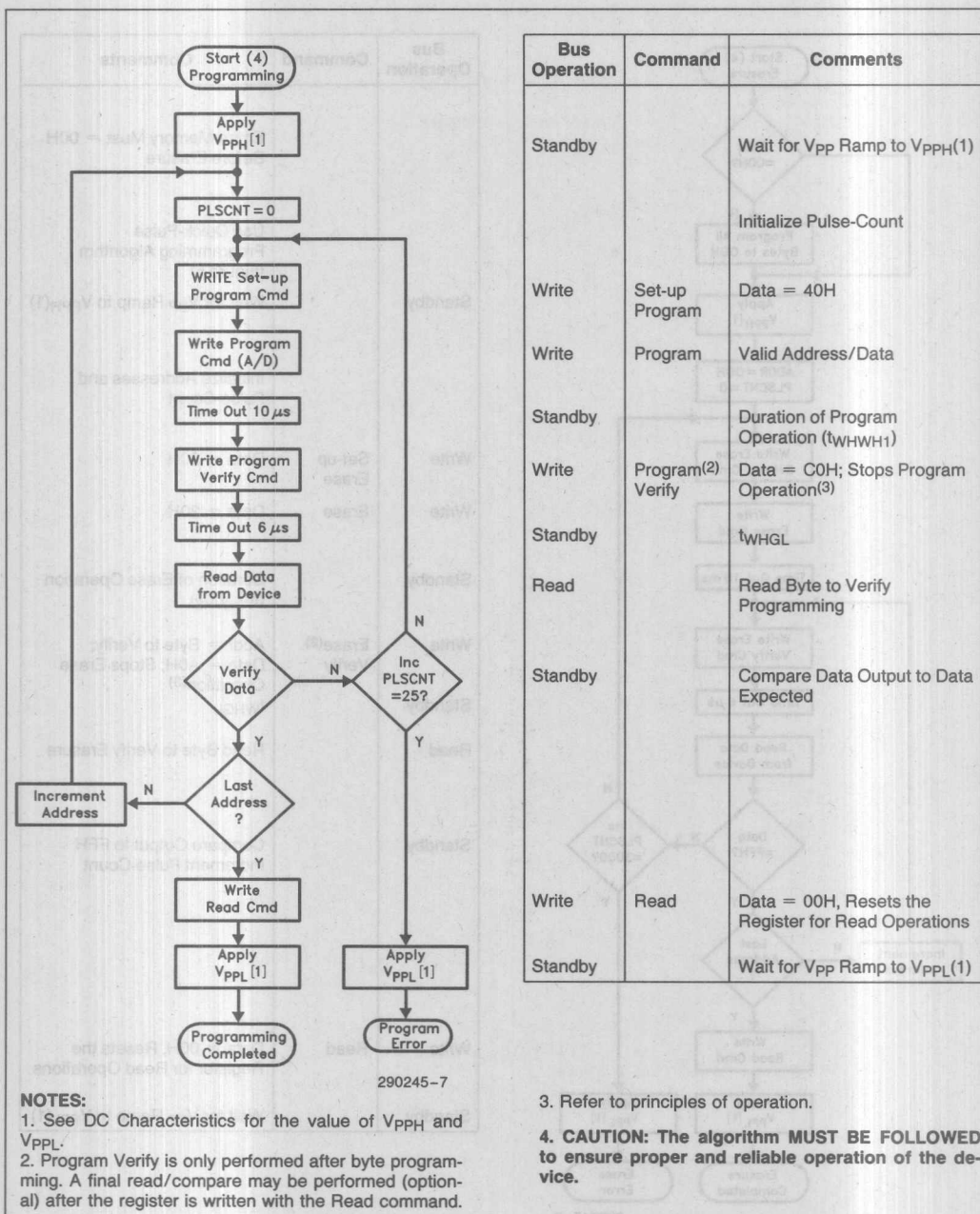


Figure 5. 28F020 Quick-Pulse Programming Algorithm

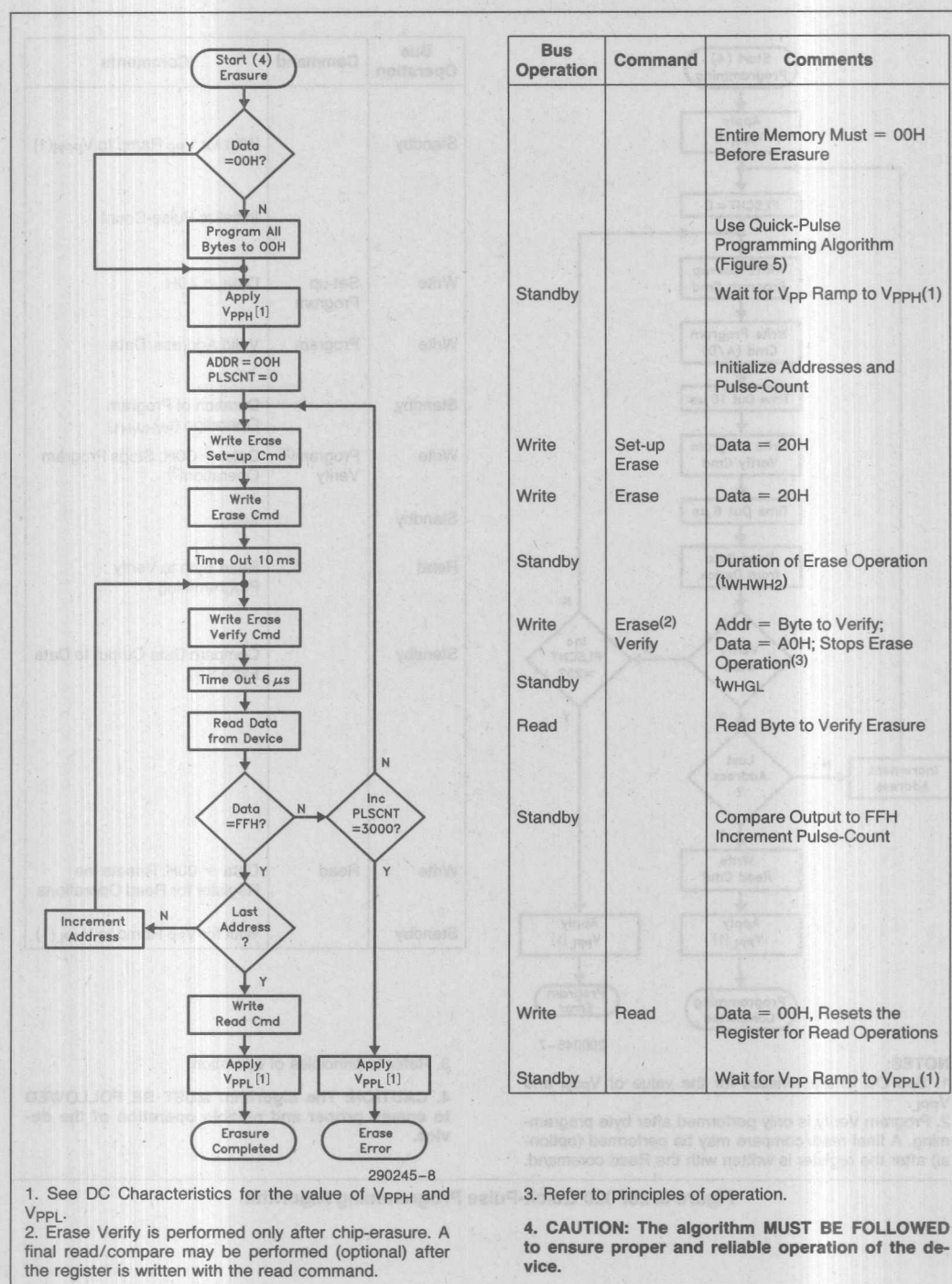


Figure 6. 28F020 Quick-Erase Algorithm

DESIGN CONSIDERATIONS

Two-Line Output Control

Flash-memories are often used in larger memory arrays. Intel provides two read-control inputs to accommodate multiple memory connections. Two-line control provides for:

- the lowest possible memory power dissipation and,
- complete assurance that output bus contention will not occur.

To efficiently use these two control inputs, an address-decoder output should drive chip-enable, while the system's read signal controls all flash-memories and other parallel memories. This assures that only enabled memory devices have active outputs, while deselected devices maintain the low power standby condition.

Power Supply Decoupling

Flash-memory power-switching characteristics require careful device decoupling. System designers are interested in three supply current (I_{CC}) issues—standby, active, and transient current peaks produced by falling and rising edges of chip-enable. The capacitive and inductive loads on the device outputs determine the magnitudes of these peaks.

Two-line control and proper decoupling capacitor selection will suppress transient voltage peaks. Each device should have a 0.1 μ F ceramic capacitor connected between V_{CC} and V_{SS} , and between V_{PP} and V_{SS} .

Place the high-frequency, low-inherent-inductance capacitors as close as possible to the devices. Also, for every eight devices, a 4.7 μ F electrolytic capacitor should be placed at the array's power supply connection, between V_{CC} and V_{SS} . The bulk capacitor will overcome voltage slumps caused by printed-circuit-board trace inductance, and will supply charge to the smaller capacitors as needed.

V_{PP} Trace on Printed Circuit Boards

Programming flash-memories, while they reside in the target system, requires that the printed circuit board designer pay attention to the V_{PP} power supply trace. The V_{PP} pin supplies the memory cell current for programming. Use similar trace widths and layout considerations given the V_{CC} power bus. Adequate V_{PP} supply traces and decoupling will decrease V_{PP} voltage spikes and overshoots.

Power Up/Down Protection

The 28F020 is designed to offer protection against accidental erasure or programming during power transitions. Upon power-up, the 28F020 is indifferent as to which power supply, V_{PP} or V_{CC} , powers up first. **Power supply sequencing is not required.** Internal circuitry in the 28F020 ensures that the command register is reset to the read mode on power up.

A system designer must guard against active writes for V_{CC} voltages above V_{LKO} when V_{PP} is active. Since both $WE\#$ and $CE\#$ must be low for a command write, driving either to V_{IH} will inhibit writes. The control register architecture provides an added level of protection since alteration of memory contents only occurs after successful completion of the two-step command sequences.

28F020 Power Dissipation

When designing portable systems, designers must consider battery power consumption not only during device operation, but also for data retention during system idle time. Flash nonvolatility increases the usable battery life of your system because the 28F020 does not consume any power to retain code or data when the system is off. Table 4 illustrates the power dissipated when updating the 28F020.

Table 4. 28F020 Typical Update Power Dissipation⁽⁴⁾

Operation	Notes	Power Dissipation (Watt-Seconds)
Array Program/Program Verify	1	0.34
Array Erase/Erase Verify	2	0.37
One Complete Cycle	3	1.05

NOTES:

- Formula to calculate typical Program/Program Verify Power = $[V_{PP} \times \# \text{ Bytes} \times \text{typical } \# \text{ Prog Pulse} (t_{WHWH1} \times I_{PP2} \text{ typical} + t_{WHGL} \times I_{PP4} \text{ typical})] + [V_{CC} \times \# \text{ Bytes} \times \text{typical } \# \text{ Prog Pulses} (t_{WHWH1} \times I_{CC2} \text{ typical} + t_{WHGL} \times I_{CC4} \text{ typical})]$.
- Formula to calculate typical Erase/Erase Verify Power = $[V_{PP} (I_{PP3} \text{ typical} \times t_{ERASE} \text{ typical} + I_{PP5} \text{ typical} \times t_{WHGL} \times \# \text{ Bytes})] + [V_{CC} (I_{CC3} \text{ typical} \times t_{ERASE} \text{ typical} + I_{CC5} \text{ typical} \times t_{WHGL} \times \# \text{ Bytes})]$.
- One Complete Cycle = Array Preprogram + Array Erase + Program.
- "Typicals" are not guaranteed but based on a limited number of samples from 28F020-150 production lots.

ABSOLUTE MAXIMUM RATINGS*

Operating Temperature	
During Read	0°C to +70°C(1)
During Erase/Program	0°C to +70°C(1)
Operating Temperature	
During Read	–40°C to +85°C(2)
During Erase/Program	–40°C to +85°C(2)
Temperature Under Bias	
	–10°C to +80°C(1)
Temperature Under Bias	
	–50°C to +95°C(2)
Storage Temperature	
	–65°C to +125°C
Voltage on Any Pin with Respect to Ground	
	–2.0V to +7.0V(2)
Voltage on Pin A ₉ with Respect to Ground	
	–2.0V to +13.5V(2, 3)
V _{PP} Supply Voltage with Respect to Ground	
During Erase/Program	–2.0V to +14.0V(2, 3)
V _{CC} Supply Voltage with Respect to Ground	
	–2.0V to +7.0V(2)
Output Short Circuit Current	
	100 mA(4)

NOTES:

1. Operating temperature is for commercial product defined by this specification.
2. Operating temperature is for extended temperature product as defined by this specification.
3. Minimum DC input voltage is –0.5V. During transitions, inputs may undershoot to –2.0V for periods less than 20 ns. Maximum DC voltage on output pins is V_{CC} + 0.5V, which may overshoot to V_{CC} + 2.0V for periods less than 20 ns.
4. Maximum DC voltage on A₉ or V_{PP} may overshoot to +14.0V for periods less than 20 ns.
5. Output shorted for no more than one second. No more than one output shorted at a time.
6. See AC Input/Output reference Waveforms and AC Testing Load Circuits for testing characteristics.
7. See High Speed AC Input/Output reference Waveforms and High Speed AC Testing Load Circuits for testing characteristics.

OPERATING CONDITIONS

Symbol	Parameter	Limits		Unit
		Min	Max	
T _A	Operating Temperature(1)	0	70	°C
T _A	Operating Temperature(2)	–40	+85	°C
V _{CC}	V _{CC} Supply Voltage (10%)(6)	4.50	5.50	V
V _{CC}	V _{CC} Supply Voltage (5%)(7)	4.75	5.25	V

DC CHARACTERISTICS—TTL/NMOS COMPATIBLE—Commercial Products

Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typ(4)	Max		
I _{LI}	Input Leakage Current	1			±1.0	μA	V _{CC} = V _{CC} Max V _{IN} = V _{CC} or V _{SS}
I _{LO}	Output Leakage Current	1			±10	μA	V _{CC} = V _{CC} Max V _{OUT} = V _{CC} or V _{SS}

NOTICE: This is a production data sheet. The specifications are subject to change without notice.

***WARNING:** Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.

DC CHARACTERISTICS—TTL/NMOS COMPATIBLE—Commercial Products (Continued)

Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typ ⁽⁴⁾	Max		
I _{CCS}	V _{CC} Standby Current	1		0.3	1.0	mA	V _{CC} = V _{CC} Max CE# = V _{IH}
I _{CC1}	V _{CC} Active Read Current	1		10	30	mA	V _{CC} = V _{CC} Max, CE# = V _{IL} f = 6 MHz, I _{OUT} = 0 mA
I _{CC2}	V _{CC} Programming Current	1, 2		1.0	10	mA	Programming in Progress
I _{CC3}	V _{CC} Erase Current	1, 2		5.0	15	mA	Erase in Progress
I _{CC4}	V _{CC} Program Verify Current	1, 2		5.0	15	mA	V _{PP} = V _{PPH} , Program Verify in Progress
I _{CC5}	V _{CC} Erase Verify Current	1, 2		5.0	15	mA	V _{PP} = V _{PPH} , Erase Verify in Progress
I _{PPS}	V _{PP} Leakage Current	1			± 10	μA	V _{PP} ≤ V _{CC}
I _{PP1}	V _{PP} Read Current, I _D Current or Standby Current	1		90	200	μA	V _{PP} > V _{CC}
					± 10	μA	V _{PP} ≤ V _{CC}
I _{PP2}	V _{PP} Programming Current	1, 2		8	30	mA	V _{PP} = V _{PPH} , Programming in Progress
I _{PP3}	V _{PP} Erase Current	1, 2		10	30	mA	V _{PP} = V _{PPH}
I _{PP4}	V _{PP} Program Verify Current	1, 2		2.0	5.0	mA	V _{PP} = V _{PPH} , Program Verify in Progress
I _{PP5}	V _{PP} Erase Verify Current	1, 2		2.0	5.0	mA	V _{PP} = V _{PPH} , Erase Verify in Progress
V _{IL}	Input Low Voltage		−0.5		0.8	V	
V _{IH}	Input High Voltage		2.0		V _{CC} + 0.5	V	
V _{OL}	Output Low Voltage				0.45	V	I _{OL} = 5.8 mA V _{CC} = V _{CC} Min
V _{OH1}	Output High Voltage		2.4			V	I _{OH} = −2.5 mA V _{CC} = V _{CC} Min
V _{ID}	A ₉ Intelligent Identifier Voltage		11.50		13.00	V	
I _{ID}	A ₉ Intelligent Identifier Current	1, 2		90	200	μA	A ₉ = V _{ID}
V _{PPL}	V _{PP} during Read-Only Operations		0.00		6.5	V	NOTE: Erase/Program are Inhibited when V _{PP} = V _{PPL}
V _{PPH}	V _{PP} during Read/Write Operations		11.40		12.60	V	
V _{LKO}	V _{CC} Erase/Write Lock Voltage		2.5			V	

DC CHARACTERISTICS—CMOS COMPATIBLE—Commercial Products

Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typ ⁽⁴⁾	Max		
I_{LI}	Input Leakage Current	1			± 1.0	μA	$V_{CC} = V_{CC} \text{ Max}$ $V_{IN} = V_{CC} \text{ or } V_{SS}$
I_{LO}	Output Leakage Current	1			± 10	μA	$V_{CC} = V_{CC} \text{ Max}$ $V_{OUT} = V_{CC} \text{ or } V_{SS}$
I_{CCS}	V_{CC} Standby Current	1		50	100	μA	$V_{CC} = V_{CC} \text{ Max}$ $CE\# = V_{CC} \pm 0.2V$
I_{CC1}	V_{CC} Active Read Current	1		10	30	mA	$V_{CC} = V_{CC} \text{ Max}$, $CE\# = V_{IL}$ $f = 6 \text{ MHz}$, $I_{OUT} = 0 \text{ mA}$
I_{CC2}	V_{CC} Programming Current	1, 2		1.0	10	mA	Programming in Progress
I_{CC3}	V_{CC} Erase Current	1, 2		5.0	15	mA	Erase in Progress
I_{CC4}	V_{CC} Program Verify Current	1, 2		5.0	15	mA	$V_{PP} = V_{PPH}$, Program Verify in Progress
I_{CC5}	V_{CC} Erase Verify Current	1, 2		5.0	15	mA	$V_{PP} = V_{PPH}$, Erase Verify in Progress
I_{PPS}	V_{PP} Leakage Current	1			± 10	μA	$V_{PP} \leq V_{CC}$
I_{PP1}	V_{PP} Read Current, ID Current or Standby Current	1		90	200	μA	$V_{PP} > V_{CC}$
					± 10		$V_{PP} \leq V_{CC}$
I_{PP2}	V_{PP} Programming Current	1, 2		8	30	mA	$V_{PP} = V_{PPH}$, Programming in Progress
I_{PP3}	V_{PP} Erase Current	1, 2		10	30	mA	$V_{PP} = V_{PPH}$, Erasure in Progress
I_{PP4}	V_{PP} Program Verify Current	1, 2		2.0	5.0	mA	$V_{PP} = V_{PPH}$, Program Verify in Progress
I_{PP5}	V_{PP} Erase Verify Current	1, 2		2.0	5.0	mA	$V_{PP} = V_{PPH}$, Erase Verify in Progress
V_{IL}	Input Low Voltage		-0.5		0.8	V	
V_{IH}	Input High Voltage		$0.7 V_{CC}$		$V_{CC} + 0.5$	V	
V_{OL}	Output Low Voltage				0.45	V	$I_{OL} = 5.8 \text{ mA}$ $V_{CC} = V_{CC} \text{ Min}$
V_{OH1}	Output High Voltage		$0.85 V_{CC}$			V	$I_{OH} = -2.5 \text{ mA}$, $V_{CC} = V_{CC} \text{ Min}$
V_{OH2}			$V_{CC} - 0.4$				$I_{OH} = -100 \mu A$, $V_{CC} = V_{CC} \text{ Min}$
V_{ID}	A_9 Intelligent Identifier Voltage		11.50		13.00	V	
I_{ID}	A_9 Intelligent Identifier Current	1, 2		90	200	μA	$A_9 = V_{ID}$
V_{PPL}	V_{PP} during Read-Only Operations		0.00		6.5	V	NOTE: Erase/Programs are Inhibited when $V_{PP} = V_{PPL}$
V_{PPH}	V_{PP} during Read/Write Operations		11.40		12.60	V	
V_{LKO}	V_{CC} Erase/Write Lock Voltage		2.5			V	

Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typ(4)	Max		
I_{LI}	Input Leakage Current	1			± 1.0	μA	$V_{CC} = V_{CC} \text{ Max}$ $V_{IN} = V_{CC} \text{ or } V_{SS}$
I_{LO}	Output Leakage Current	1			± 10	μA	$V_{CC} = V_{CC} \text{ Max}$ $V_{OUT} = V_{CC} \text{ or } V_{SS}$
I_{CCS}	V_{CC} Standby Current	1		0.3	1.0	mA	$V_{CC} = V_{CC} \text{ Max}$ $CE\# = V_{IH}$
I_{CC1}	V_{CC} Active Read Current	1		10	30	mA	$V_{CC} = V_{CC} \text{ Max}$, $CE\# = V_{IL}$ $f = 6 \text{ MHz}$, $I_{OUT} = 0 \text{ mA}$
I_{CC2}	V_{CC} Programming Current	1, 2		1.0	30	mA	Programming in Progress
I_{CC3}	V_{CC} Erase Current	1, 2		5.0	30	mA	Erase in Progress
I_{CC4}	V_{CC} Program Verify Current	1, 2		5.0	30	mA	$V_{PP} = V_{PPH}$, Program Verify in Progress
I_{CC5}	V_{CC} Erase Verify Current	1, 2		5.0	30	mA	$V_{PP} = V_{PPH}$, Erase Verify in Progress
I_{PPS}	V_{PP} Leakage Current	1			± 10	μA	$V_{PP} \leq V_{CC}$
I_{PP1}	V_{PP} Read Current, ID Current or Standby Current	1		90	200	μA	$V_{PP} > V_{CC}$
					± 10		$V_{PP} \leq V_{CC}$
I_{PP2}	V_{PP} Programming Current	1, 2		8	30	mA	$V_{PP} = V_{PPH}$ Programming in Progress
I_{PP3}	V_{PP} Erase Current	1, 2		10	30	mA	$V_{PP} = V_{PPH}$
I_{PP4}	V_{PP} Program Verify Current	1, 2		2.0	5.0	mA	$V_{PP} = V_{PPH}$, Program Verify in Progress
I_{PP5}	V_{PP} Erase Verify Current	1, 2		2.0	5.0	mA	$V_{PP} = V_{PPH}$, Erase Verify in Progress
V_{IL}	Input Low Voltage		-0.5		0.8	V	
V_{IH}	Input High Voltage		2.0		$V_{CC} + 0.5$	V	
V_{OL}	Output Low Voltage				0.45	V	$I_{OL} = 5.8 \text{ mA}$ $V_{CC} = V_{CC} \text{ Min}$
V_{OH1}	Output High Voltage		2.4			V	$I_{OH} = -2.5 \text{ mA}$ $V_{CC} = V_{CC} \text{ Min}$
V_{ID}	A_9 Intelligent Identifier Voltage		11.50		13.00	V	
I_{ID}	A_9 Intelligent Identifier Current	1, 2		90	500	μA	$A_9 = V_{ID}$
V_{PPL}	V_{PP} during Read-Only Operations		0.00		6.5	V	NOTE: Erase/Program are Inhibited when $V_{PP} = V_{PPL}$
V_{PPH}	V_{PP} during Read/Write Operations		11.40		12.60	V	
V_{LKO}	V_{CC} Erase/Write Lock Voltage		2.5			V	

DC CHARACTERISTICS—CMOS COMPATIBLE—Extended Temperature Products

Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typ ⁽⁴⁾	Max		
I_{LI}	Input Leakage Current	1			± 1.0	μA	$V_{CC} = V_{CC} \text{ Max}$ $V_{IN} = V_{CC} \text{ or } V_{SS}$
I_{LO}	Output Leakage Current	1			± 10	μA	$V_{CC} = V_{CC} \text{ Max}$ $V_{OUT} = V_{CC} \text{ or } V_{SS}$
I_{CCS}	V_{CC} Standby Current	1		50	100	μA	$V_{CC} = V_{CC} \text{ Max}$ $CE\# = V_{CC} \pm 0.2V$
I_{CC1}	V_{CC} Active Read Current	1		10	50	mA	$V_{CC} = V_{CC} \text{ Max}$, $CE\# = V_{IL}$ $f = 6 \text{ MHz}$, $I_{OUT} = 0 \text{ mA}$
I_{CC2}	V_{CC} Programming Current	1, 2		1.0	10	mA	Programming in Progress
I_{CC3}	V_{CC} Erase Current	1, 2		5.0	30	mA	Erase in Progress
I_{CC4}	V_{CC} Program Verify Current	1, 2		5.0	30	mA	$V_{PP} = V_{PPH}$, Program Verify in Progress
I_{CC5}	V_{CC} Erase Verify Current	1, 2		5.0	30	mA	$V_{PP} = V_{PPH}$, Erase Verify in Progress
I_{PPS}	V_{PP} Leakage Current	1			± 10	μA	$V_{PP} \leq V_{CC}$
I_{PP1}	V_{PP} Read Current, ID Current or Standby Current	1		90	200	μA	$V_{PP} > V_{CC}$
					± 10		$V_{PP} \leq V_{CC}$
I_{PP2}	V_{PP} Programming Current	1, 2		8	30	mA	$V_{PP} = V_{PPH}$, Programming in Progress
I_{PP3}	V_{PP} Erase Current	1, 2		10	30	mA	$V_{PP} = V_{PPH}$, Erase in Progress
I_{PP4}	V_{PP} Program Verify Current	1, 2		2.0	5.0	mA	$V_{PP} = V_{PPH}$, Program Verify in Progress
I_{PP5}	V_{PP} Erase Verify Current	1, 2		2.0	5.0	mA	$V_{PP} = V_{PPH}$, Erase Verify in Progress
V_{IL}	Input Low Voltage		-0.5		0.8	V	
V_{IH}	Input High Voltage		$0.7 V_{CC}$		$V_{CC} + 0.5$	V	
V_{OL}	Output Low Voltage				0.45	V	$I_{OL} = 5.8 \text{ mA}$ $V_{CC} = V_{CC} \text{ Min}$
V_{OH1}	Output High Voltage		$0.85 V_{CC}$			V	$I_{OH} = -2.5 \text{ mA}$, $V_{CC} = V_{CC} \text{ Min}$
V_{OH2}							$I_{OH} = -100 \mu A$, $V_{CC} = V_{CC} \text{ Min}$
V_{ID}	A_9 Intelligent Identifier Voltage		11.50		13.00	V	
I_{ID}	A_9 Intelligent Identifier Current	1, 2		90	500	μA	$A_9 = V_{ID}$
V_{PPL}	V_{PP} during Read-Only Operations		0.00		6.5	V	NOTE: Erase/Programs are Inhibited when $V_{PP} = V_{PPL}$
V_{PPH}	V_{PP} during Read/Write Operations		11.40		12.60	V	
V_{LKO}	V_{CC} Erase/Write Lock Voltage		2.5			V	

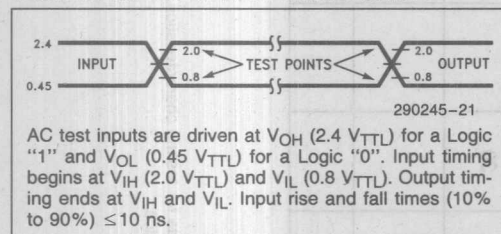
CAPACITANCE $T_A = 25^\circ\text{C}$, $f = 1.0\text{ MHz}$

Symbol	Parameter	Notes	Limits		Unit	Conditions
			Min	Max		
C_{IN}	Address/Control Capacitance	3		8	pF	$V_{IN} = 0V$
C_{OUT}	Output Capacitance	3		12	pF	$V_{OUT} = 0V$

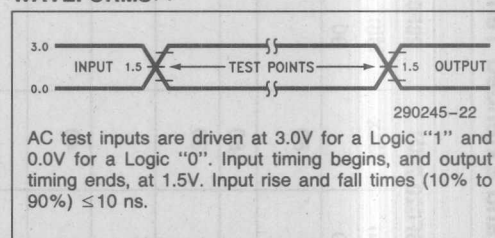
NOTES for DC Characteristics and Capacitance:

1. All currents are in RMS unless otherwise noted. Typical values at $V_{CC} = 5.0V$, $V_{PP} = 12.0V$, $T = 25^\circ\text{C}$. These currents are valid for all product versions (packages and speeds).
2. Not 100% tested: Characterization data available.
3. Sampled, not 100% tested.
4. "Typicals" are not guaranteed, but based on a limited number of samples from production lots.

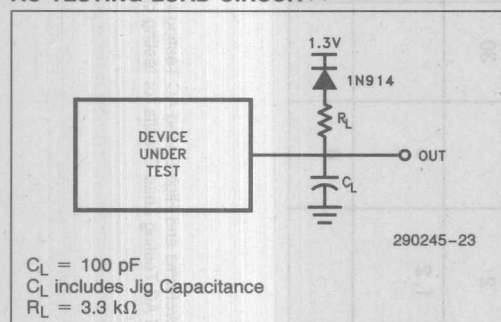
TESTING INPUT/OUTPUT WAVEFORM(1)



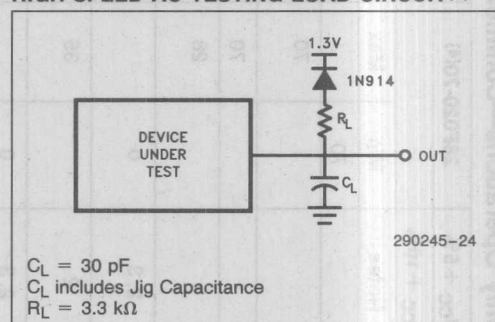
HIGH SPEED AC TESTING INPUT/OUTPUT WAVEFORMS(2)



AC TESTING LOAD CIRCUIT(1)



HIGH SPEED AC TESTING LOAD CIRCUIT(2)



AC TEST CONDITIONS(1)

Input Rise and Fall Times (10% to 90%) 10 ns
 Input Pulse Levels 0.45 and 2.4
 Input Timing Reference Level 0.8 and 2.0
 Output Timing Reference Level 0.8 and 2.0
 Capacitive Load C_L 100 pF

HIGH SPEED AC TEST CONDITIONS(2)

Input Rise and Fall Times (10% to 90%) 10 ns
 Input Pulse Levels 0.0 and 3.0
 Input Timing Reference Level 1.5
 Output Timing Reference Level 1.5
 Capacitive Load C_L 30 pF

NOTES:

1. Testing characteristics for 28F020-70 in standard configuration, and 28F020-90 and 28F020-150.
2. Testing characteristics for 28F020-70 in high speed configuration.

AC CHARACTERISTICS—Read Only Operations—Commercial and Extended Temperature Products

28F020

Versions		V _{CC} + 5%	28F020-70 ⁽⁴⁾								Unit
		V _{CC} + 10%			28F020-70 ⁽⁵⁾		28F020-90 ⁽⁵⁾		28F020-150 ⁽⁵⁾		
Symbol	Characteristics	Notes	Min	Max	Min	Max	Min	Max	Min	Max	
t _{AVAV} /t _{RC}	Read Cycle Time		70		80		90		150		ns
t _{ELQV} /t _{CE}	Chip Enable Access Time			70		80		90		120	ns
t _{AVQV} /t _{ACC}	Address Access Time			70		80		90		120	ns
t _{GLQV} /t _{OE}	Output Enable Access Time			28		30		35		50	ns
t _{ELQX} /t _{LZ}	Chip Enable to Output in Low Z	2, 3	0		0						ns
t _{EHQZ}	Chip Disable to Output in High Z	2		35		40		45		55	ns
t _{GLQX} /t _{OLZ}	Output Enable to Output in Low Z	2, 3	0		0			0			ns
t _{GHQZ} /t _{DF}	Output Disable to Output in High Z	2		30		30		30		30	ns
t _{OH}	Output Hold from Address, CE #, or OE # Change	1, 2	0		0			0			ns
t _{WHGL}	Write Recovery Time before Read		6		6			6		6	μs

NOTES:

1. Whichever occurs first.
2. Sampled, not 100% tested.
3. Guaranteed by design.
4. See High Speed AC Input/Output reference Waveforms and High Speed AC Testing Load Circuits for testing characteristics.
5. See AC Input/Output reference Waveforms and AC Testing Load Circuits for testing characteristics.

5-21



AC CHARACTERISTICS—Write/Erase/Program Only Operations⁽¹⁾—Commercial and Extended Temperature Products

Versions		V _{CC} + 5%	28F020-70 ^(2, 4)							
		V _{CC} + 10%			28F020-70 ⁽⁵⁾		28F020-90 ⁽⁵⁾		28F020-150 ⁽⁵⁾	
Symbol	Characteristics	Notes	Min	Max	Min	Max	Min	Max	Min	Max
t _{AVAV} /t _{WC}	Write Cycle Time		70		80		90		150	
t _{AVWL} /t _{AS}	Address Set-Up Time		0		0		0		0	
t _{WLAX} /t _{AH}	Address Hold Time		40		40		40		40	
		6					55			
t _{DVWH} /t _{DS}	Data Set-Up Time		40		40		40		40	
		6					55			
t _{WHDX} /t _{DH}	Data Hold Time		10		10		10		10	
t _{WHGL}	Write Recovery Time before Read		6		6		6		6	
t _{GHWL}	Read Recovery Time before Write	2	0		0		0		0	
t _{ELWL} /t _{CS}	Chip Enable Set-Up Time before Write		15		15		15		15	
t _{WHEH} /t _{CH}	Chip Enable Hold Time		0		0		0		0	
t _{WLWH} /t _{WP}	Write Pulse Width		40		40		40		60	
		6					55			
t _{WHWL} /t _{WPH}	Write Pulse Width High		20		20		20		20	
t _{WHWH1}	Duration of Programming Operation	3	10		10		10		10	
t _{WHWH2}	Duration of Erase Operation	3	9.5		9.5		9.5		9.5	
t _{VPEL}	V _{PP} Set-Up Time to Chip Enable Low	2	1		1		1		1	

NOTES:

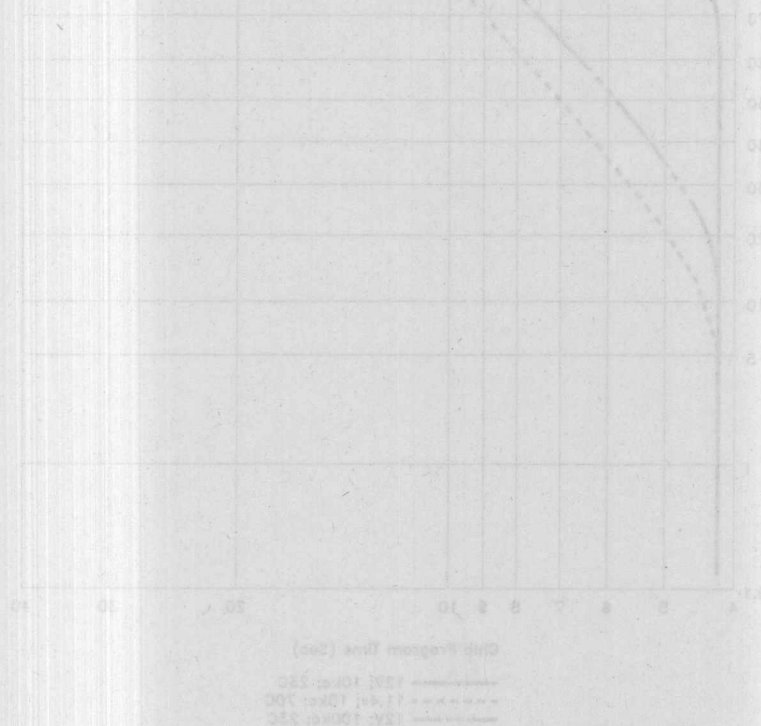
1. Read timing characteristics during read/write operations are the same as during read-only operations. Refer to AC Characteristics for Read-Only Operations.
2. Guaranteed by design.
3. The integrated stop timer terminates the programming/erase operations, thus eliminating the need for a maximum specification.
4. See High Speed AC Input/Output reference Waveforms and High Speed AC Testing Load Circuits for testing characteristics.
5. See AC Input/Output reference Waveforms and AC Testing Load Circuits for testing characteristics.
6. Minimum Specification for Extended Temperature product.

ERASE AND PROGRAMMING PERFORMANCE

Parameter	Notes	Limits			Unit
		Min	Typ	Max	
Chip Erase Time	1, 3, 4		2	30	Sec
Chip Program Time	1, 2, 4		4	25	Sec

NOTES:

1. "Typicals" are not guaranteed, but based on a limited number of samples from production lots. Data taken at 25°C, 12.0V V_{pp} at 0 cycles.
2. Minimum byte programming time excluding system overhead is 16 μsec (10 μsec program + 6 μsec write recovery), while maximum is 400 μsec/byte (16 μsec x 25 loops allowed by algorithm). Max chip programming time is specified lower than the worst case allowed by the programming algorithm since most bytes program significantly faster than the worst case byte.
3. Excludes 00H Programming prior to Erasure.
4. Excludes System-Level Overhead.
5. Refer to RR-60, 69 "ETOX Flash Memory Reliability Data Summaries" for typical cycling data and failure rate calculations.



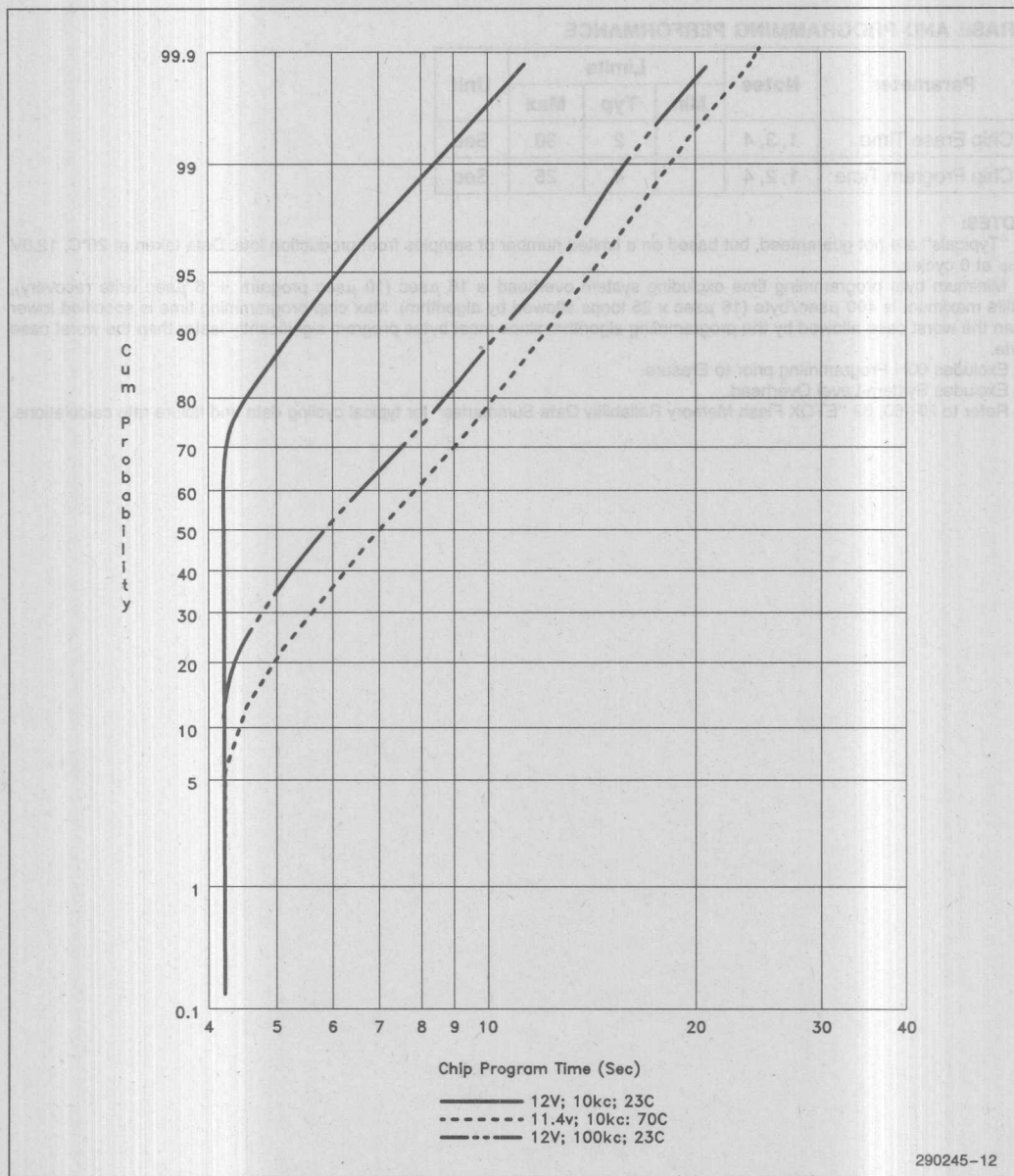


Figure 8. 28F020 Typical Programming Capability

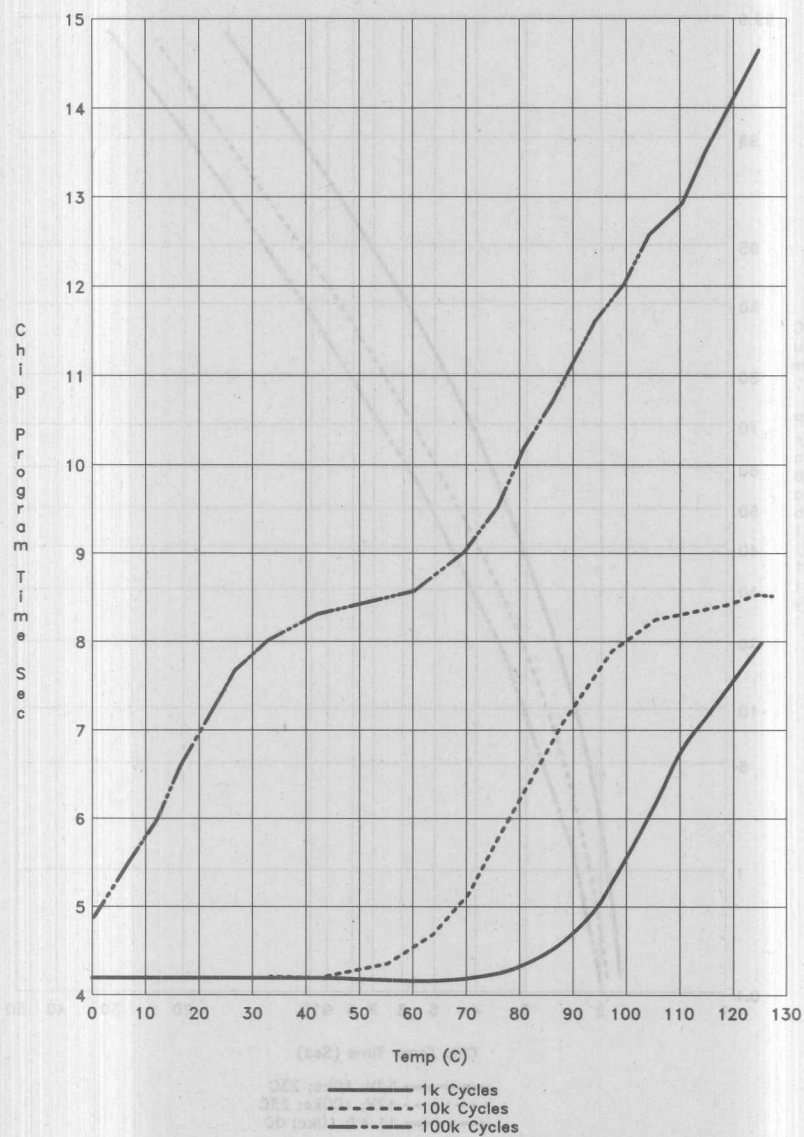
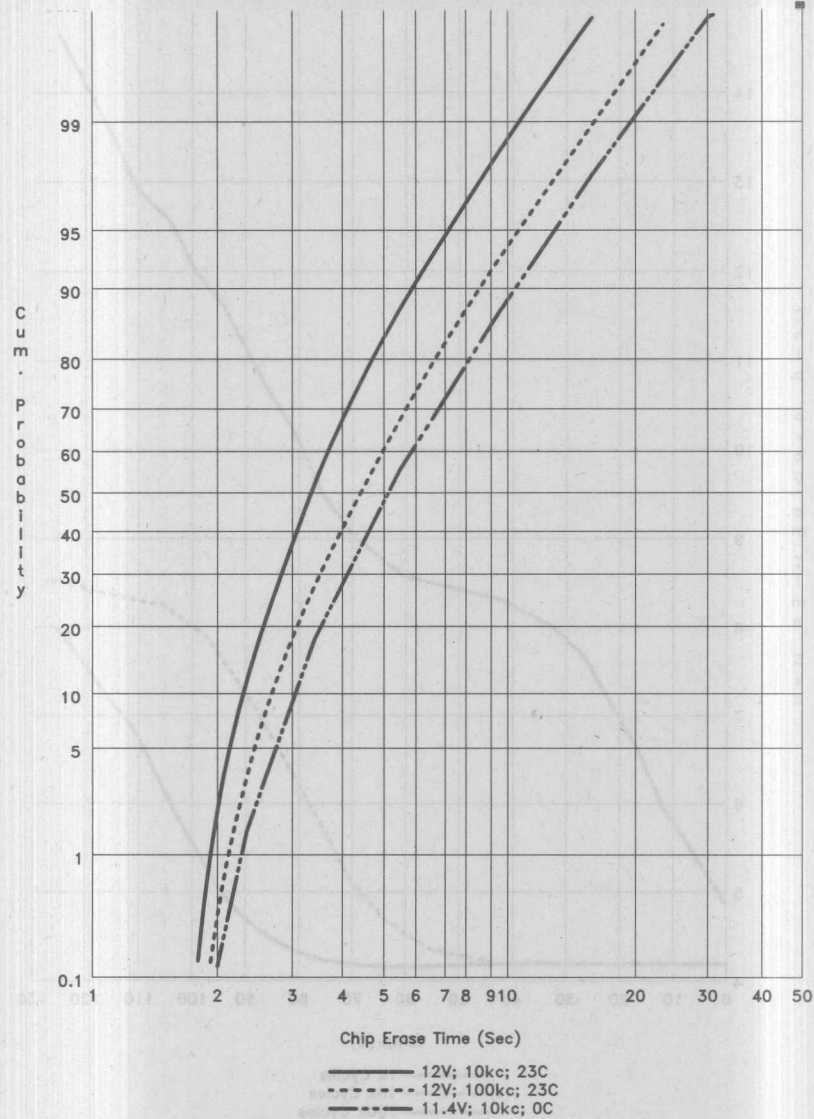
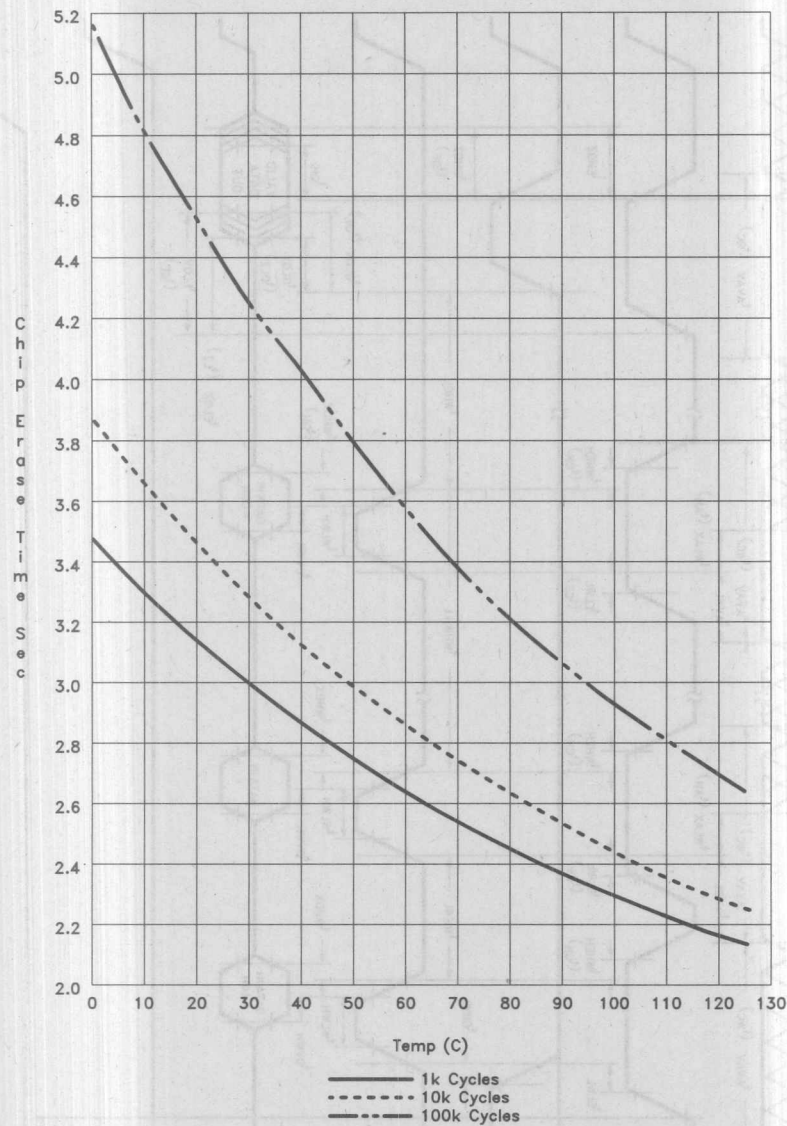


Figure 9. 28F020 Typical Program Time at 12V



NOTE:
Does not include Pre-Erase Program.

Figure 10. 28F020 Typical Erase Capability



NOTE:
Does not include Pre-Erase Program.

290245-15

Figure 11. 28F020 Typical Erase Time at 12.0V

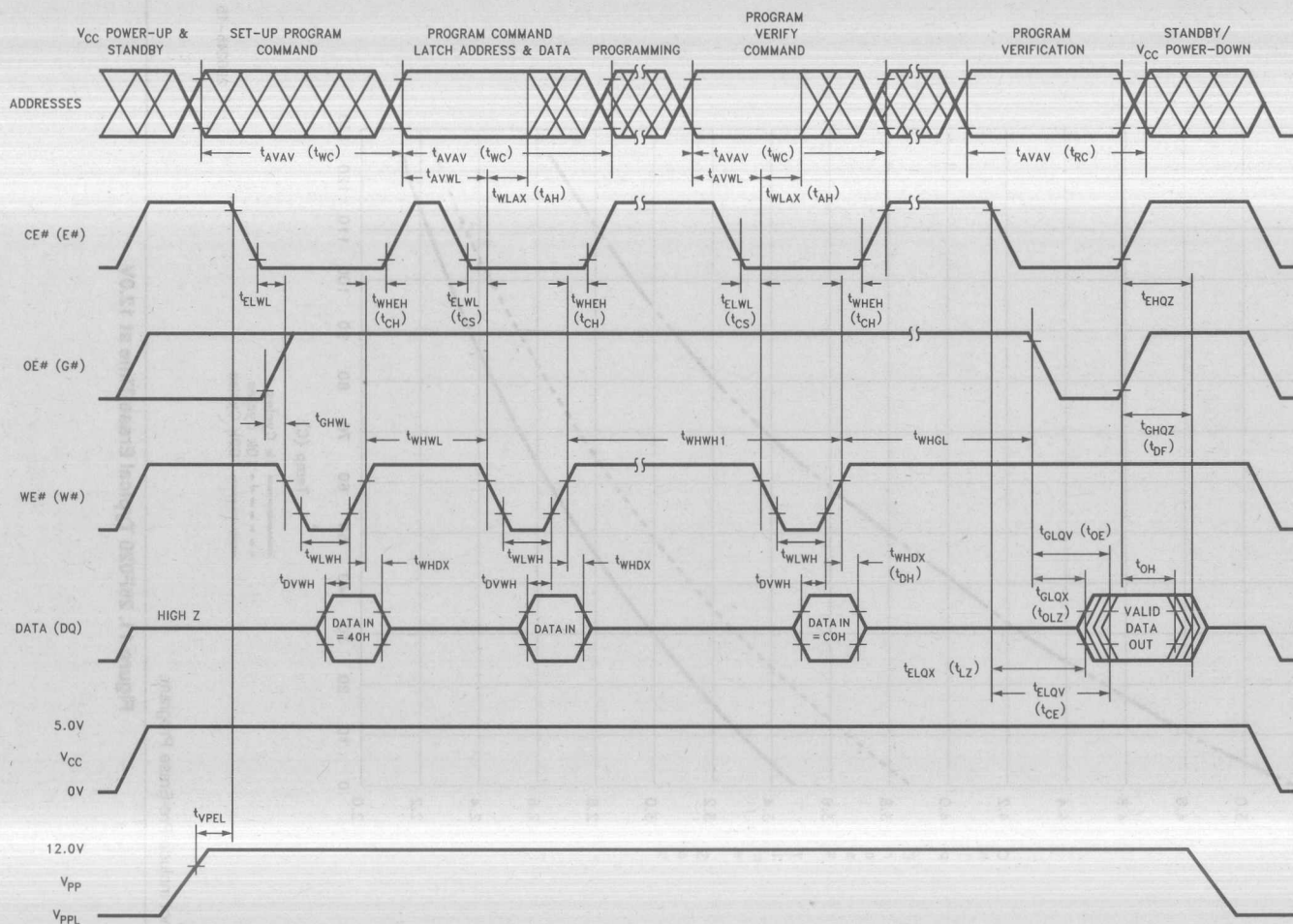
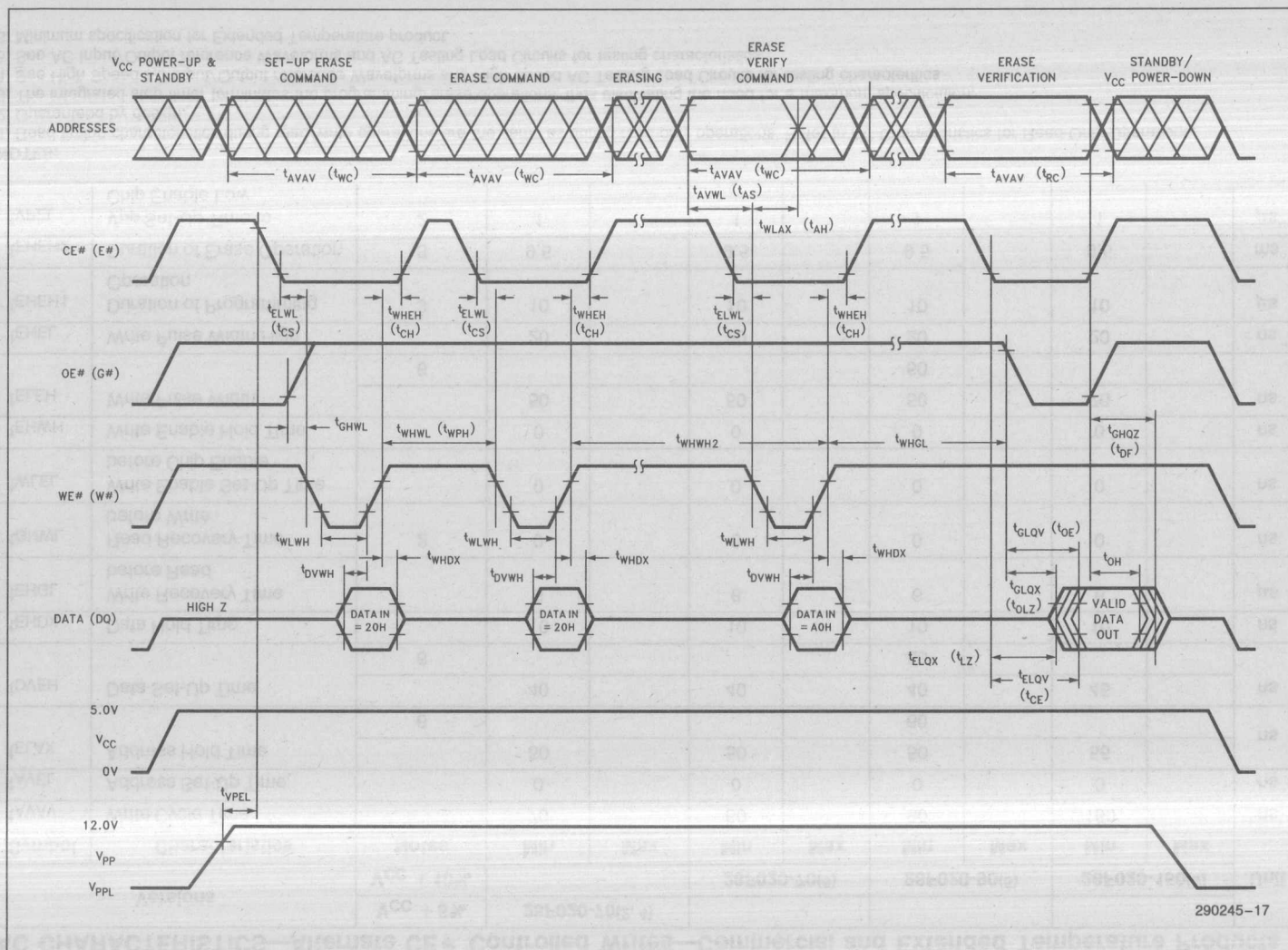


Figure 13. AC Waveforms for Erase Operations



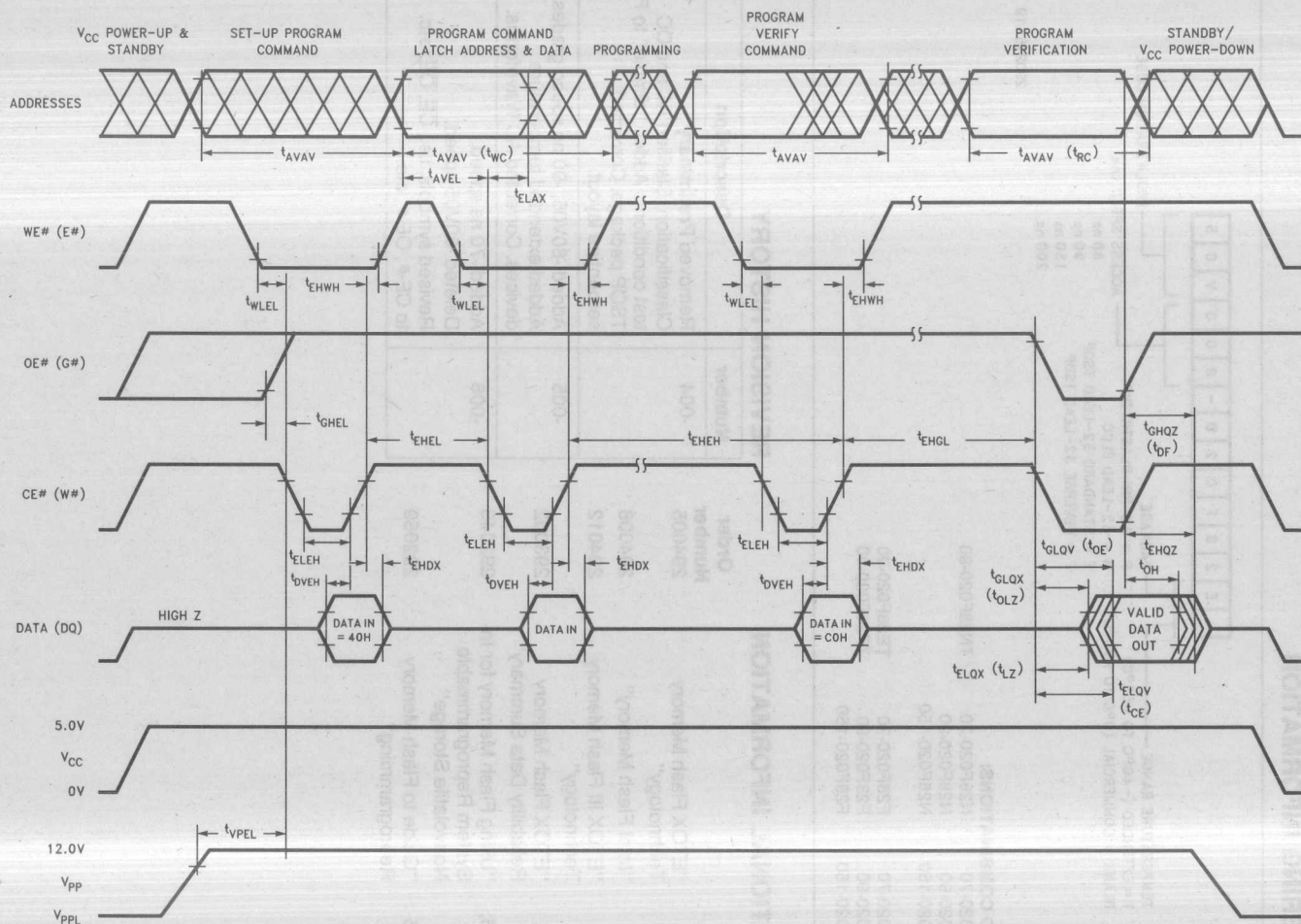
290245-17

AC CHARACTERISTICS—Alternate CE # Controlled Writes—Commercial and Extended Temperature Products

Versions		V _{CC} + 5%	28F020-70(2, 4)								Unit
		V _{CC} + 10%			28F020-70(5)		28F020-90(5)		28F020-150(5)		
Symbol	Characteristics	Notes	Min	Max	Min	Max	Min	Max	Min	Max	
t _{AVAV}	Write Cycle Time		70		80		90		150		ns
t _{AVEL}	Address Set-Up Time		0		0		0		0		ns
t _{ELAX}	Address Hold Time		50		50		50		55		ns
		6					60				
t _{DVEH}	Data Set-Up Time		40		40		40		45		ns
		6					50				
t _{EHDX}	Data Hold Time		10		10		10		10		ns
t _{EHGL}	Write Recovery Time before Read		6		6		6		6		μs
t _{GHWL}	Read Recovery Time before Write	2	0		0		0		0		ns
t _{WLEL}	Write Enable Set-Up Time before Chip Enable		0		0		0		0		ns
t _{EHWH}	Write Enable Hold Time		0		0		0		0		ns
t _{ELEH}	Write Pulse Width		50		50		50		70		ns
		6					60				
t _{EHEL}	Write Pulse Width High		20		20		20		20		ns
t _{EHEH1}	Duration of Programming Operation	3	10		10		10		10		μs
t _{EHEH2}	Duration of Erase Operation	3	9.5		9.5		9.5		9.5		ms
t _{VPEL}	V _{pp} Set-Up Time to Chip Enable Low	2	1		1		1		1		μs

NOTES:

1. Read timing characteristics during read/write operations are the same as during read-only operations. Refer to AC Characteristics for Read-Only Operations.
2. Guaranteed by design.
3. The integrated stop timer terminates the programming/erase operations, thus eliminating the need for a maximum specification.
4. See High Speed AC Input/Output reference Waveforms and High Speed AC Testing Load Circuits for testing characteristics.
5. See AC Input/Output reference Waveforms and AC Testing Load Circuits for testing characteristics.
6. Minimum specification for Extended Temperature product.



NOTE:
Alternative CE-Controlled Write Timings also apply to erase operations.

Figure 14. Alternate AC Waveforms for Programming Operations

ORDERING INFORMATION

E	2	8	F	0	2	0	-	8	0	0	V	0	5
---	---	---	---	---	---	---	---	---	---	---	---	---	---

TEMPERATURE RANGE

T = EXTENDED (-40°C TO +55°C)

BLANK = COMMERCIAL (0°C TO +70°C)

PACKAGE

P = 32-PIN PLASTIC DIP

N = 32-LEAD PLCC

E = STANDARD 32-LEAD TSOP

F = REVERSE 32-LEAD TSOP

PROLIFERATION CODE

ACCESS SPEED (ns)

80 ns

90 ns

150 ns

200 ns

290245-19

VALID COMBINATIONS:

P28F020-70

N28F020-70

TN28F020-90

P28F020-90

N28F020-90

P28F020-150

N28F020-150

E28F020-70

F28F020-70

TE28F020-90

E28F020-90

F28F020-90

TF28F020-90

E28F020-150

F28F020-150

ADDITIONAL INFORMATION

		Order Number
ER-20,	"ETOX Flash Memory Technology"	294005
ER-24,	"Intel Flash Memory"	294008
ER-28,	"ETOX III Flash Memory Technology"	294012
RR-60,	"ETOX Flash Memory Reliability Data Summary"	293002
AP-316,	"Using Flash Memory for In-System Reprogrammable Nonvolatile Storage"	292046
AP-325	"Guide to Flash Memory Reprogramming"	292059

REVISION HISTORY

Number	Description
-004	Removed Preliminary Classification. Clarified AC and DC test conditions. Added "dimple" to F TSOP package. Corrected serpentine layout.
-005	Added -80V05, -90 ns speed grades. Added extended temperature devices. Corrected AC Waveforms.
-006	Added -70 ns speed. Deleted -80V05 speed. Revised symbols, i.e., \overline{CE} , \overline{OE} , etc. to CE#, OE#, etc.

28F010

1024K (128K x 8) CMOS FLASH MEMORY

- **Flash Electrical Chip-Erase**
 - 1 Second Typical Chip-Erase
- **Quick Pulse Programming Algorithm**
 - 10 μ s Typical Byte-Program
 - 2 Second Chip-Program
- **100,000 Erase/Program Cycles**
- **12.0V \pm 5% V_{PP}**
- **High-Performance Read**
 - 65 ns Maximum Access Time
- **CMOS Low Power Consumption**
 - 10 mA Typical Active Current
 - 50 μ A Typical Standby Current
 - 0 Watts Data Retention Power
- **Integrated Program/Erase Stop Timer**
- **Command Register Architecture for Microprocessor/Microcontroller Compatible Write Interface**
- **Noise Immunity Features**
 - \pm 10% V_{CC} Tolerance
 - Maximum Latch-Up Immunity through EPI Processing
- **ETOX Nonvolatile Flash Technology**
 - EPROM-Compatible Process Base
 - High-Volume Manufacturing Experience
- **JEDEC-Standard Pinouts**
 - 32-Pin Plastic Dip
 - 32-Lead PLCC
 - 32-Lead TSOP

(See Packaging Spec., Order #231369)

■ Extended Temperature Options

Intel's 28F010 CMOS flash memory offers the most cost-effective and reliable alternative for read/write random access nonvolatile memory. The 28F010 adds electrical chip-erase and reprogramming to familiar EPROM technology. Memory contents can be rewritten: in a test socket; in a PROM-programmer socket; on-board during subassembly test; in-system during final test; and in-system after-sale. The 28F010 increases memory flexibility, while contributing to time and cost savings.

The 28F010 is a 1024 kilobit nonvolatile memory organized as 131,072 bytes of 8 bits. Intel's 28F010 is offered in 32-pin plastic dip or 32-lead PLCC and TSOP packages. Pin assignments conform to JEDEC standards for byte-wide EPROMs.

Extended erase and program cycling capability is designed into Intel's ETOX (EPROM Tunnel Oxide) process technology. Advanced oxide processing, an optimized tunneling structure, and lower electric field combine to extend reliable cycling beyond that of traditional EEPROMs. With the 12.0V V_{PP} supply, the 28F010 performs 100,000 erase and program cycles well within the time limits of the Quick Pulse Programming and Quick Erase algorithms.

Intel's 28F010 employs advanced CMOS circuitry for systems requiring high-performance access speeds, low power consumption, and immunity to noise. Its 65 nanosecond access time provides no-WAIT-state performance for a wide range of microprocessors and microcontrollers. Maximum standby current of 100 μ A translates into power savings when the device is deselected. Finally, the highest degree of latch-up protection is achieved through Intel's unique EPI processing. Prevention of latch-up is provided for stresses up to 100 mA on address and data pins, from -1 V to V_{CC} + 1V.

With Intel's ETOX process base, the 28F010 levers years of EPROM experience to yield the highest levels of quality, reliability, and cost-effectiveness.

NO INTERNAL CONNECTION TO DEVICE. Pin must be driven or left floating.	NC
GROUND	V _{SS}
DEVICE POWER SUPPLY (2V \pm 10%)	V _{CC}
ERASE/PROGRAM POWER SUPPLY for writing (10V \pm 5%)	V _{PP}
Note: With V _{PP} < 6.5V, memory contents can only be erased.	

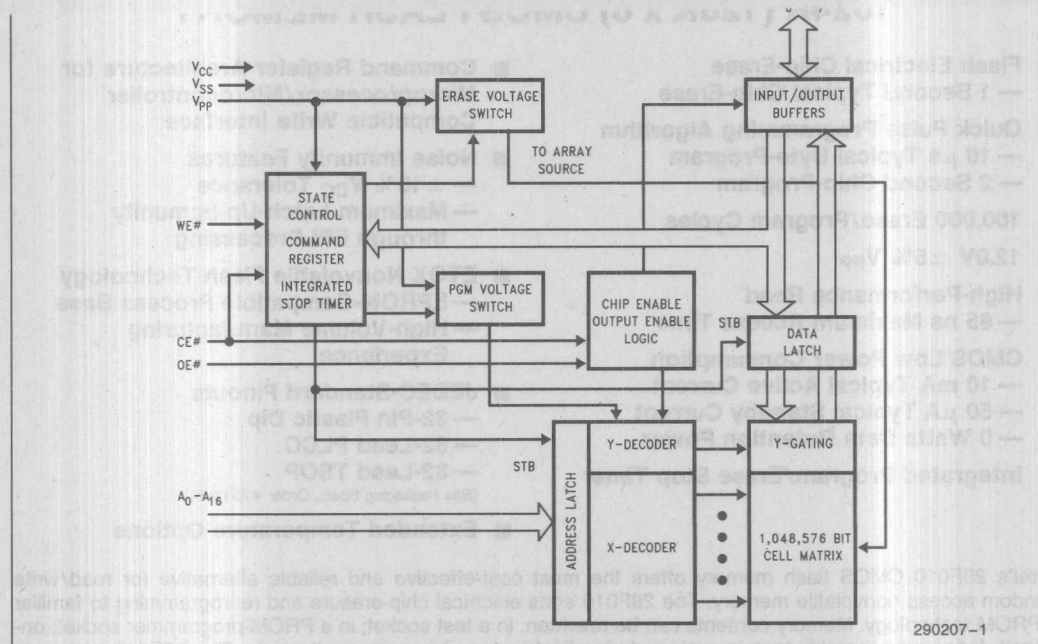


Figure 1. 28F010 Block Diagram

Table 1. Pin Description

Symbol	Type	Name and Function
A ₀ -A ₁₆	INPUT	ADDRESS INPUTS for memory addresses. Addresses are internally latched during a write cycle.
DQ ₀ -DQ ₇	INPUT/OUTPUT	DATA INPUT/OUTPUT: Inputs data during memory write cycles; outputs data during memory read cycles. The data pins are active high and float to tri-state OFF when the chip is deselected or the outputs are disabled. Data is internally latched during a write cycle.
CE #	INPUT	CHIP ENABLE: Activates the device's control logic, input buffers, decoders and sense amplifiers. CE # is active low; CE # high deselects the memory device and reduces power consumption to standby levels.
OE #	INPUT	OUTPUT ENABLE: Gates the devices output through the data buffers during a read cycle. OE # is active low.
WE #	INPUT	WRITE ENABLE: Controls writes to the control register and the array. Write enable is active low. Addresses are latched on the falling edge and data is latched on the rising edge of the WE # pulse. Note: With V _{PP} ≤ 6.5V, memory contents cannot be altered.
V _{PP}		ERASE/PROGRAM POWER SUPPLY for writing the command register, erasing the entire array, or programming bytes in the array.
V _{CC}		DEVICE POWER SUPPLY (5V ± 10%)
V _{SS}		GROUND
NC		NO INTERNAL CONNECTION to device. Pin may be driven or left floating.

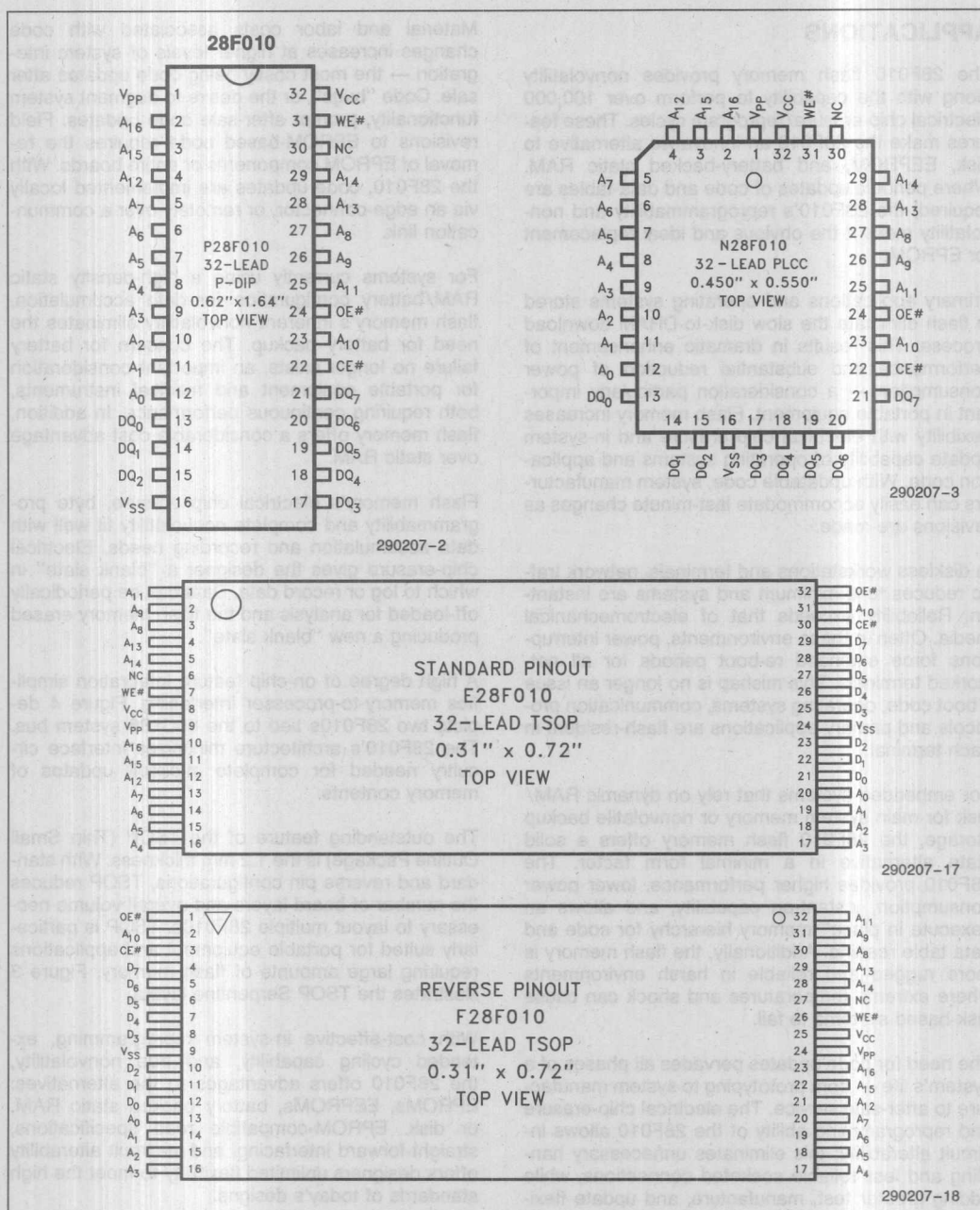


Figure 2. 28F010 Pin Configurations

APPLICATIONS

The 28F010 flash memory provides nonvolatility along with the capability to perform over 100,000 electrical chip-erase/reprogram cycles. These features make the 28F010 an innovative alternative to disk, EEPROM, and battery-backed static RAM. Where periodic updates of code and data-tables are required, the 28F010's reprogrammability and nonvolatility make it the obvious and ideal replacement for EPROM.

Primary applications and operating systems stored in flash eliminate the slow disk-to-DRAM download process. This results in dramatic enhancement of performance and substantial reduction of power consumption — a consideration particularly important in portable equipment. Flash memory increases flexibility with electrical chip erasure and in-system update capability of operating systems and application code. With updatable code, system manufacturers can easily accommodate last-minute changes as revisions are made.

In diskless workstations and terminals, network traffic reduces to a minimum and systems are instant-on. Reliability exceeds that of electromechanical media. Often in these environments, power interruptions force extended re-boot periods for all networked terminals. This mishap is no longer an issue if boot code, operating systems, communication protocols and primary applications are flash-resident in each terminal.

For embedded systems that rely on dynamic RAM/disk for main system memory or nonvolatile backup storage, the 28F010 flash memory offers a solid state alternative in a minimal form factor. The 28F010 provides higher performance, lower power consumption, instant-on capability, and allows an "execute in place" memory hierarchy for code and data table reading. Additionally, the flash memory is more rugged and reliable in harsh environments where extreme temperatures and shock can cause disk-based systems to fail.

The need for code updates pervades all phases of a system's life — from prototyping to system manufacture to after-sale service. The electrical chip-erase and reprogramming ability of the 28F010 allows in-circuit alterability; this eliminates unnecessary handling and less-reliable socketed connections, while adding greater test, manufacture, and update flexibility.

Material and labor costs associated with code changes increases at higher levels of system integration — the most costly being code updates after sale. Code "bugs", or the desire to augment system functionality, prompt after-sale code updates. Field revisions to EPROM-based code requires the removal of EPROM components or entire boards. With the 28F010, code updates are implemented locally via an edge-connector, or remotely over a communication link.

For systems currently using a high-density static RAM/battery configuration for data accumulation, flash memory's inherent nonvolatility eliminates the need for battery backup. The concern for battery failure no longer exists, an important consideration for portable equipment and medical instruments, both requiring continuous performance. In addition, flash memory offers a considerable cost advantage over static RAM.

Flash memory's electrical chip erasure, byte programmability and complete nonvolatility fit well with data accumulation and recording needs. Electrical chip-erase gives the designer a "blank slate" in which to log or record data. Data can be periodically off-loaded for analysis and the flash memory erased producing a new "blank slate".

A high degree of on-chip feature integration simplifies memory-to-processor interfacing. Figure 4 depicts two 28F010s tied to the 80C186 system bus. The 28F010's architecture minimizes interface circuitry needed for complete in-circuit updates of memory contents.

The outstanding feature of the TSOP (Thin Small Outline Package) is the 1.2 mm thickness. With standard and reverse pin configurations, TSOP reduces the number of board layers and overall volume necessary to layout multiple 28F010s. TSOP is particularly suited for portable equipment and applications requiring large amounts of flash memory. Figure 3 illustrates the TSOP Serpentine layout.

With cost-effective in-system reprogramming, extended cycling capability, and true nonvolatility, the 28F010 offers advantages to the alternatives: EPROMs, EEPROMs, battery backed static RAM, or disk. EPROM-compatible read specifications, straight-forward interfacing, and in-circuit alterability offers designers unlimited flexibility to meet the high standards of today's designs.

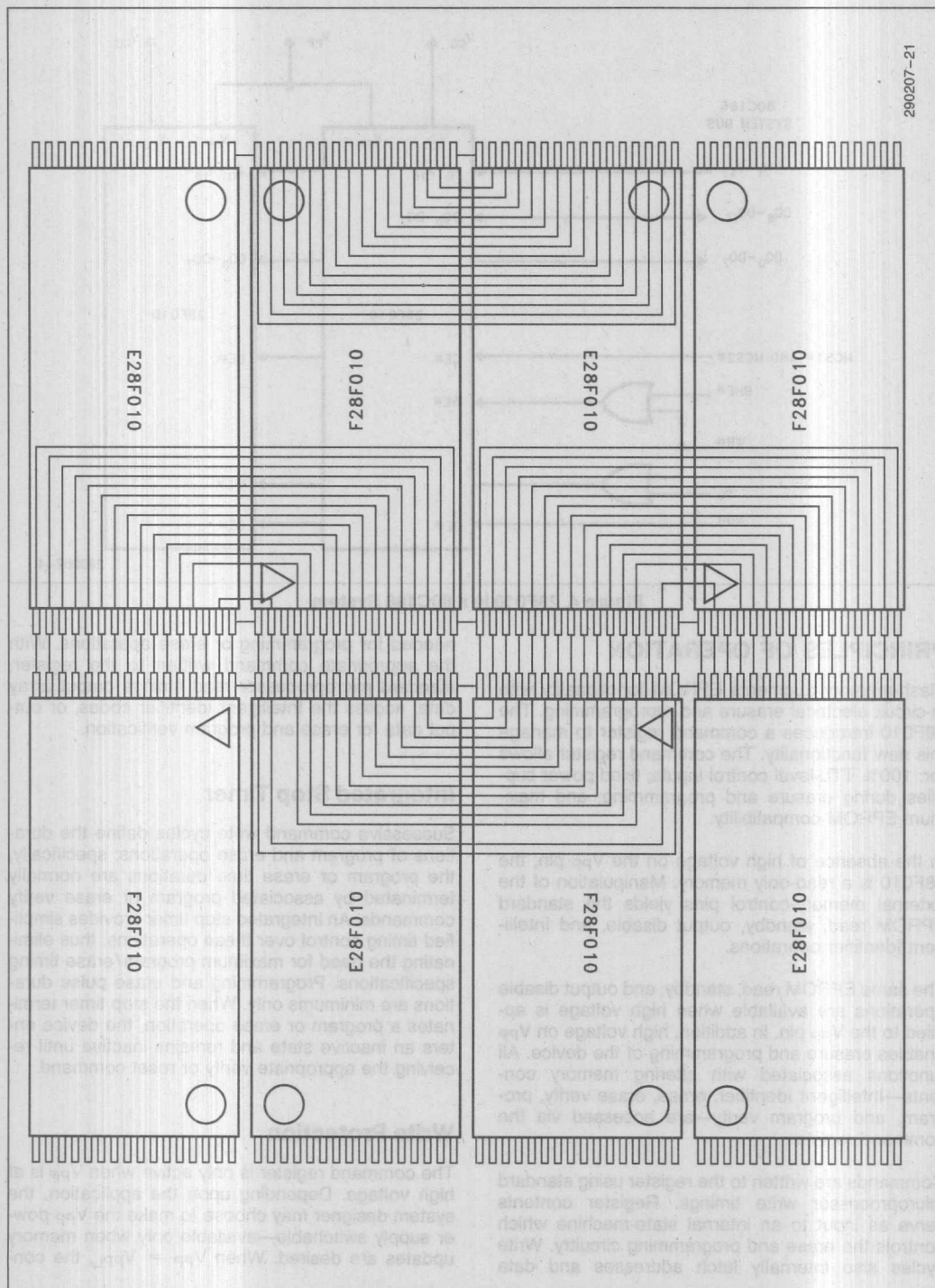


Figure 3. TSOP Serpentine Layout

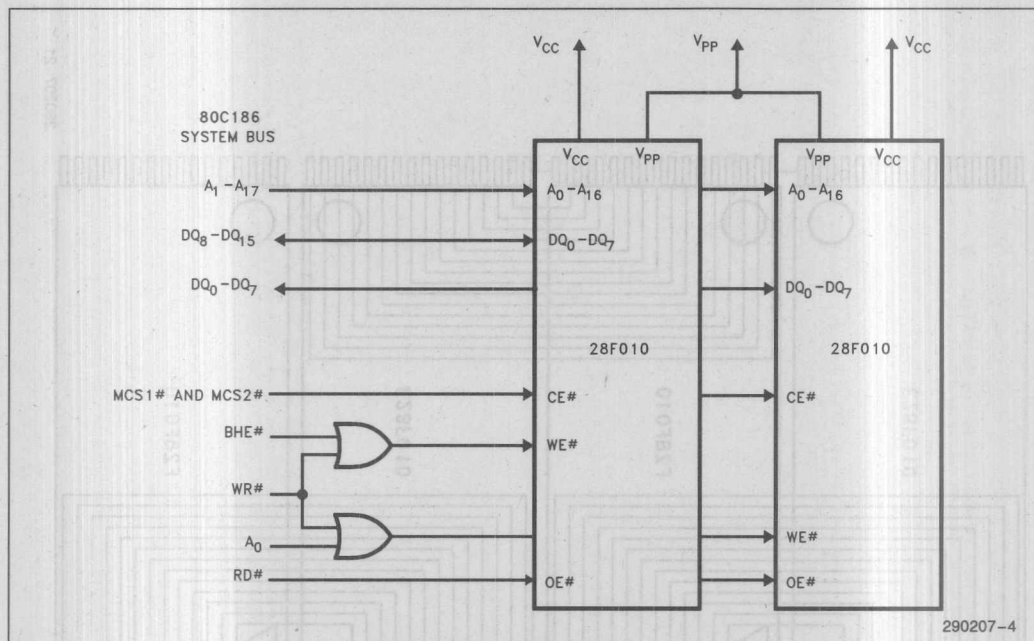


Figure 4. 28F010 in a 80C186 System

PRINCIPLES OF OPERATION

Flash-memory augments EPROM functionality with in-circuit electrical erasure and reprogramming. The 28F010 introduces a command register to manage this new functionality. The command register allows for: 100% TTL-level control inputs; fixed power supplies during erasure and programming; and maximum EPROM compatibility.

In the absence of high voltage on the V_{pp} pin, the 28F010 is a read-only memory. Manipulation of the external memory-control pins yields the standard EPROM read, standby, output disable, and Intelligent Identifier operations.

The same EPROM read, standby, and output disable operations are available when high voltage is applied to the V_{pp} pin. In addition, high voltage on V_{pp} enables erasure and programming of the device. All functions associated with altering memory contents—Intelligent Identifier, erase, erase verify, program, and program verify—are accessed via the command register.

Commands are written to the register using standard microprocessor write timings. Register contents serve as input to an internal state-machine which controls the erase and programming circuitry. Write cycles also internally latch addresses and data

needed for programming or erase operations. With the appropriate command written to the register, standard microprocessor read timings output array data, access the Intelligent Identifier codes, or output data for erase and program verification.

Integrated Stop Timer

Successive command write cycles define the durations of program and erase operations; specifically, the program or erase time durations are normally terminated by associated program or erase verify commands. An integrated stop timer provides simplified timing control over these operations; thus eliminating the need for maximum program/erase timing specifications. Programming and erase pulse durations are minimums only. When the stop timer terminates a program or erase operation, the device enters an inactive state and remains inactive until receiving the appropriate verify or reset command.

Write Protection

The command register is only active when V_{pp} is at high voltage. Depending upon the application, the system designer may choose to make the V_{pp} power supply switchable—available only when memory updates are desired. When $V_{pp} = V_{ppl}$, the con-

Table 2. 28F010 Bus Operations

		Pins	V _{PP} (1)	A ₀	A ₉	CE #	OE #	WE #	DQ ₀ –DQ ₇
Operation									
READ-ONLY	Read		V _{PPL}	A ₀	A ₉	V _{IL}	V _{IL}	V _{IH}	Data Out
	Output Disable		V _{PPL}	X	X	V _{IL}	V _{IH}	V _{IH}	Tri-State
	Standby		V _{PPL}	X	X	V _{IH}	X	X	Tri-State
	Intelligent Identifier (Mfr) ⁽²⁾		V _{PPL}	V _{IL}	V _{ID} ⁽³⁾	V _{IL}	V _{IL}	V _{IH}	Data = 89H
	Intelligent Identifier (Device) ⁽²⁾		V _{PPL}	V _{IH}	V _{ID} ⁽³⁾	V _{IL}	V _{IL}	V _{IH}	Data = B4H
READ/WRITE	Read		V _{PPH}	A ₀	A ₉	V _{IL}	V _{IL}	V _{IH}	Data Out ⁽⁴⁾
	Output Disable		V _{PPH}	X	X	V _{IL}	V _{IH}	V _{IH}	Tri-State
	Standby ⁽⁵⁾		V _{PPH}	X	X	V _{IH}	X	X	Tri-State
	Write		V _{PPH}	A ₀	A ₉	V _{IL}	V _{IH}	V _{IL}	Data In ⁽⁶⁾

NOTES:

1. Refer to DC Characteristics. When V_{PP} = V_{PPL} memory contents can be read but not written or erased.
2. Manufacturer and device codes may also be accessed via a command register write sequence. Refer to Table 3. All other addresses low.
3. V_{ID} is the Intelligent Identifier high voltage. Refer to DC Characteristics.
4. Read operations with V_{PP} = V_{PPH} may access array data or the Intelligent Identifier codes.
5. With V_{PP} at high voltage, the standby current equals I_{CC} + I_{PP} (standby).
6. Refer to Table 3 for valid Data-In during a write operation.
7. X can be V_{IL} or V_{IH}.

tents of the register default to the read command, making the 28F010 a read-only memory. In this mode, the memory contents cannot be altered.

Or, the system designer may choose to “hardwire” V_{PP}, making the high voltage supply constantly available. In this case, all Command Register functions are inhibited whenever V_{CC} is below the write lockout voltage V_{LKO}. (See Power Up/Down Protection) The 28F010 is designed to accommodate either design practice, and to encourage optimization of the processor-memory interface.

The two-step program/erase write sequence to the Command Register provides additional software write protections.

BUS OPERATIONS**Read**

The 28F010 has two control functions, both of which must be logically active, to obtain data at the outputs. Chip-Enable (CE#) is the power control and should be used for device selection. Output-Enable (OE#) is the output control and should be used to gate data from the output pins, independent of device selection. Refer to AC read timing waveforms.

When V_{PP} is high (V_{PPH}), the read operation can be used to access array data, to output the Intelligent Identifier codes, and to access data for program/

erase verification. When V_{PP} is low (V_{PPL}), the read operation can **only** access the array data.

Output Disable

With Output-Enable at a logic-high level (V_{IH}), output from the device is disabled. Output pins are placed in a high-impedance state.

Standby

With Chip-Enable at a logic-high level, the standby operation disables most of the 28F010's circuitry and substantially reduces device power consumption. The outputs are placed in a high-impedance state, independent of the Output-Enable signal. If the 28F010 is deselected during erasure, programming, or program/erase verification, the device draws active current until the operation is terminated.

Intelligent Identifier Operation

The Intelligent Identifier operation outputs the manufacturer code (89H) and device code (B4H). Programming equipment automatically matches the device with its proper erase and programming algorithms.

With Chip-Enable and Output-Enable at a logic low level, raising A9 to high voltage V_{ID} (see DC Characteristics) activates the operation. Data read from locations 0000H and 0001H represent the manufacturer's code and the device code, respectively.

The manufacturer- and device-codes can also be read via the command register, for instances where the 28F010 is erased and reprogrammed in the target system. Following a write of 90H to the command register, a read from address location 0000H outputs the manufacturer code (89H). A read from address 0001H outputs the device code (B4H).

Write

Device erasure and programming are accomplished via the command register, when high voltage is applied to the V_{PP} pin. The contents of the register serve as input to the internal state-machine. The state-machine outputs dictate the function of the device.

The command register itself does not occupy an addressable memory location. The register is a latch

used to store the command, along with address and data information needed to execute the command.

The command register is written by bringing Write-Enable to a logic-low level (V_{IL}), while Chip-Enable is low. Addresses are latched on the falling edge of Write-Enable, while data is latched on the rising edge of the Write-Enable pulse. Standard microprocessor write timings are used.

Refer to AC Write Characteristics and the Erase/Programming Waveforms for specific timing parameters.

COMMAND DEFINITIONS

When low voltage is applied to the V_{PP} pin, the contents of the command register default to 00H, enabling read-only operations.

Placing high voltage on the V_{PP} pin enables read/write operations. Device operations are selected by writing specific data patterns into the command register. Table 3 defines these 28F010 register commands.

Table 3. Command Definitions

Command	Bus Cycles Req'd	First Bus Cycle			Second Bus Cycle		
		Operation(1)	Address(2)	Data(3)	Operation(1)	Address(2)	Data(3)
Read Memory	1	Write	X	00H			
Read Intelligent Identifier Codes(4)	3	Write	X	90H	Read	(4)	(4)
Set-up Erase/Erase(5)	2	Write	X	20H	Write	X	20H
Erase Verify(5)	2	Write	EA	A0H	Read	X	EVD
Set-up Program/Program(6)	2	Write	X	40H	Write	PA	PD
Program Verify(6)	2	Write	X	C0H	Read	X	PVD
Reset(7)	2	Write	X	FFH	Write	X	FFH

NOTES:

- Bus operations are defined in Table 2.
- IA = Identifier address: 00H for manufacturer code, 01H for device code.
EA = Address of memory location to be read during erase verify.
PA = Address of memory location to be programmed.
Addresses are latched on the falling edge of the Write-Enable pulse.
- ID = Data read from location IA during device identification (Mfr = 89H, Device = B4H).
EVD = Data read from location EA during erase verify.
PD = Data to be programmed at location PA. Data is latched on the rising edge of Write Enable.
PVD = Data read from location PA during program verify. PA is latched on the Program command.
- Following the Read intelligent ID command, two read operations access manufacturer and device codes.
- Figure 6 illustrates the Quick Erase Algorithm.
- Figure 5 illustrates the Quick Pulse Programming Algorithm.
- The second bus cycle must be followed by the desired command register write.

Read Command

While V_{pp} is high, for erasure and programming, memory contents can be accessed via the read command. The read operation is initiated by writing 00H into the command register. Microprocessor read cycles retrieve array data. The device remains enabled for reads until the command register contents are altered.

The default contents of the register upon V_{pp} power-up is 00H. This default value ensures that no spurious alteration of memory contents occurs during the V_{pp} power transition. Where the V_{pp} supply is hard-wired to the 28F010, the device powers-up and remains enabled for reads until the command-register contents are changed. Refer to the AC Read Characteristics and Waveforms for specific timing parameters.

Intelligent Identifier Command

Flash memories are intended for use in applications where the local CPU alters memory contents. As such, manufacturer- and device-codes must be accessible while the device resides in the target system. PROM programmers typically access signature codes by raising A9 to a high voltage. However, multiplexing high voltage onto address lines is not a desired system-design practice.

The 28F010 contains an Intelligent Identifier operation to supplement traditional PROM-programming methodology. The operation is initiated by writing 90H into the command register. Following the command write, a read cycle from address 0000H retrieves the manufacturer code of 89H. A read cycle from address 0001H returns the device code of B4H. To terminate the operation, it is necessary to write another valid command into the register.

Set-up Erase/Erase Commands

Set-up Erase is a command-only operation that stages the device for electrical erasure of all bytes in the array. The set-up erase operation is performed by writing 20H to the command register.

To commence chip-erasure, the erase command (20H) must again be written to the register. The erase operation begins with the rising edge of the Write-Enable pulse and terminates with the rising edge of the next Write-Enable pulse (i.e., Erase-Verify Command).

This two-step sequence of set-up followed by execution ensures that memory contents are not accidentally erased. Also, chip-erasure can only occur when high voltage is applied to the V_{pp} pin. In the absence

of this high voltage, memory contents are protected against erasure. Refer to AC Erase Characteristics and Waveforms for specific timing parameters.

Erase-Verify Command

The erase command erases all bytes of the array in parallel. After each erase operation, all bytes must be verified. The erase verify operation is initiated by writing A0H into the command register. The address for the byte to be verified must be supplied as it is latched on the falling edge of the Write-Enable pulse. The register write terminates the erase operation with the rising edge of its Write-Enable pulse.

The 28F010 applies an internally-generated margin voltage to the addressed byte. Reading FFH from the addressed byte indicates that all bits in the byte are erased.

The erase-verify command must be written to the command register prior to each byte verification to latch its address. The process continues for each byte in the array until a byte does not return FFH data, or the last address is accessed.

In the case where the data read is not FFH, another erase operation is performed. (Refer to Set-up Erase/Erase). Verification then resumes from the address of the last-verified byte. Once all bytes in the array have been verified, the erase step is complete. The device can be programmed. At this point, the verify operation is terminated by writing a valid command (e.g. Program Set-up) to the command register. Figure 6, the Quick Erase algorithm, illustrates how commands and bus operations are combined to perform electrical erasure of the 28F010. Refer to AC Erase Characteristics and Waveforms for specific timing parameters.

Set-up Program/Program Commands

Set-up program is a command-only operation that stages the device for byte programming. Writing 40H into the command register performs the set-up operation.

Once the program set-up operation is performed, the next Write-Enable pulse causes a transition to an active programming operation. Addresses are internally latched on the falling edge of the Write-Enable pulse. Data is internally latched on the rising edge of the Write-Enable pulse. The rising edge of Write-Enable also begins the programming operation. The programming operation terminates with the next rising edge of Write-Enable, used to write the program-verify command. Refer to AC Programming Characteristics and Waveforms for specific timing parameters.

Program-verify Command

The 28F010 is programmed on a byte-by-byte basis. Byte programming may occur sequentially or at random. Following each programming operation, the byte just programmed must be verified.

The program-verify operation is initiated by writing C0H into the command register. The register write terminates the programming operation with the rising edge of its Write-Enable pulse. The program-verify operation stages the device for verification of the byte last programmed. No new address information is latched.

The 28F010 applies an internally-generated margin voltage to the byte. A microprocessor read cycle outputs the data. A successful comparison between the programmed byte and true data means that the byte is successfully programmed. Programming then proceeds to the next desired byte location. Figure 5, the 28F010 Quick Pulse Programming algorithm, illustrates how commands are combined with bus operations to perform byte programming. Refer to AC Programming Characteristics and Waveforms for specific timing parameters.

Reset Command

A reset command is provided as a means to safely abort the erase- or program-command sequences. Following either set-up command (erase or program) with two consecutive writes of FFH will safely abort the operation. Memory contents will not be altered. A valid command must then be written to place the device in the desired state.

EXTENDED ERASE/PROGRAM CYCLING

EEPROM cycling failures have always concerned users. The high electrical field required by thin oxide EEPROMs for tunneling can literally tear apart the oxide at defect regions. To combat this, some suppliers have implemented redundancy schemes, reducing cycling failures to insignificant levels. However, redundancy requires that cell size be doubled—an expensive solution.

Intel has designed extended cycling capability into its ETOX flash memory technology. Resulting improvements in cycling reliability come without increasing memory cell size or complexity. First, an advanced tunnel oxide increases the charge carrying ability ten-fold. Second, the oxide area per cell subjected to the tunneling electric field is one-tenth that of common EEPROMs, minimizing the probability of oxide defects in the region. Finally, the peak electric field during erasure is approximately 2 MV/cm lower than EEPROM. The lower electric

field greatly reduces oxide stress and the probability of failure—increasing time to wearout by a factor of 100,000,000.

The 28F010 is capable of 100,000 program/erase cycles. The device is programmed and erased using Intel's Quick Pulse Programming and Quick Erase algorithms. Intel's algorithmic approach uses a series of operations (pulses), along with byte verification, to completely and reliably erase and program the device.

For further information, see Reliability Report RR-60.

QUICK PULSE PROGRAMMING ALGORITHM

The Quick Pulse Programming algorithm uses programming operations of 10 μ s duration. Each operation is followed by a byte verification to determine when the addressed byte has been successfully programmed. The algorithm allows for up to 25 programming operations per byte, although most bytes verify on the first or second operation. The entire sequence of programming and byte verification is performed with V_{pp} at high voltage. Figure 5 illustrates the Quick Pulse Programming algorithm.

QUICK ERASE ALGORITHM

Intel's Quick Erase algorithm yields fast and reliable electrical erasure of memory contents. The algorithm employs a closed-loop flow, similar to the Quick Pulse Programming algorithm, to simultaneously remove charge from all bits in the array.

Erasure begins with a read of memory contents. The 28F010 is erased when shipped from the factory. Reading FFH data from the device would immediately be followed by device programming.

For devices being erased and reprogrammed, uniform and reliable erasure is ensured by first programming all bits in the device to their charged state (Data = 00H). This is accomplished, using the Quick Pulse Programming algorithm, in approximately two seconds.

Erase execution then continues with an initial erase operation. Erase verification (data = FFH) begins at address 0000H and continues through the array to the last address, or until data other than FFH is encountered. With each erase operation, an increasing number of bytes verify to the erased state. Erase efficiency may be improved by storing the address of the last byte verified in a register. Following the next erase operation, verification starts at that stored address location. Erasure typically occurs in one second. Figure 6 illustrates the Quick Erase algorithm.

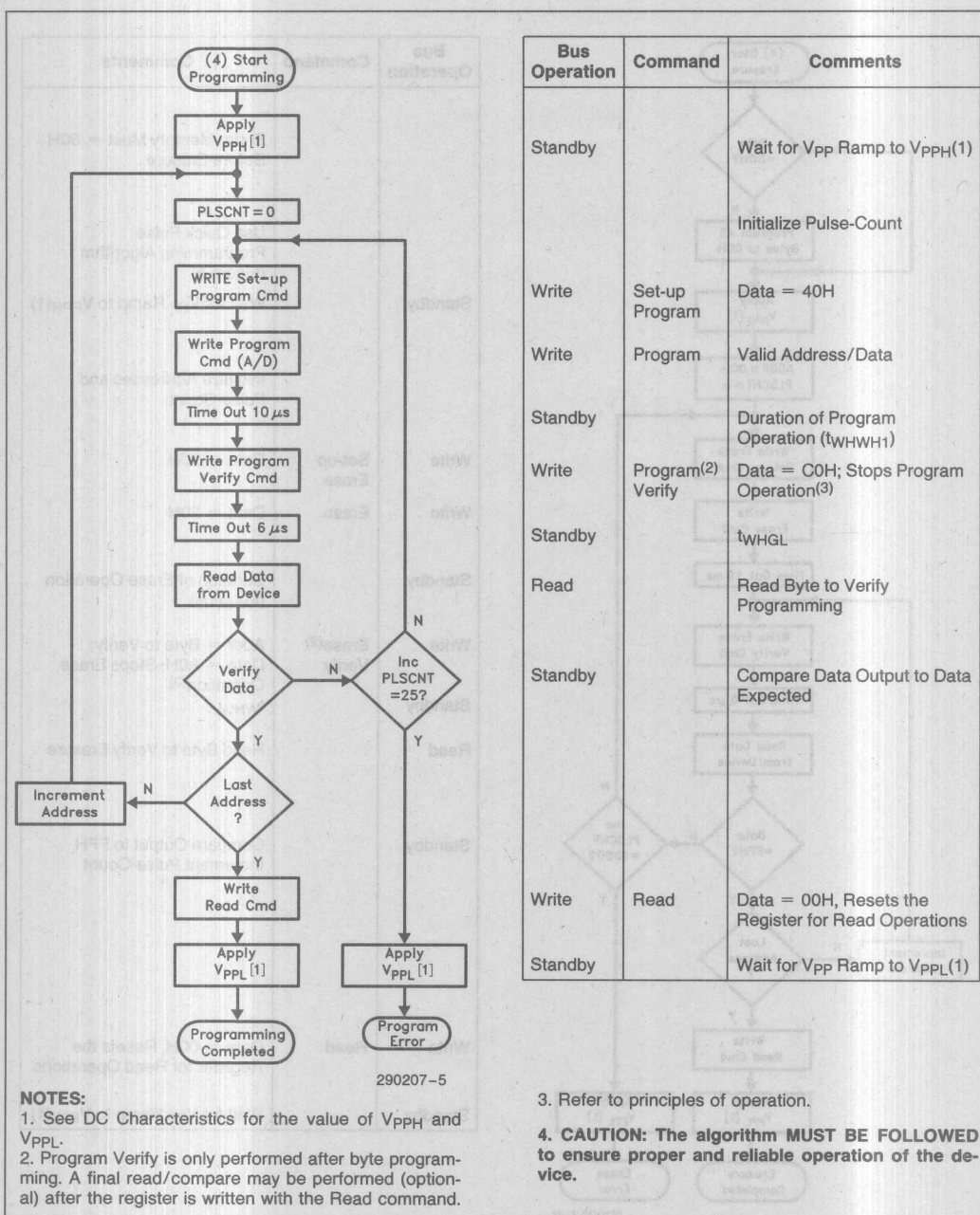


Figure 5. 28F010 Quick Pulse Programming Algorithm

Bus Operation	Command	Comments
Standby		Wait for V_{pp} Ramp to $V_{ppH}(1)$
		Initialize Pulse-Count
Write	Set-up Program	Data = 40H
Write	Program	Valid Address/Data
Standby		Duration of Program Operation (t_{WHWH1})
Write	Program(2) Verify	Data = C0H; Stops Program Operation(3)
Standby		t_{WHGL}
Read		Read Byte to Verify Programming
Standby		Compare Data Output to Data Expected
Write	Read	Data = 00H, Resets the Register for Read Operations
Standby		Wait for V_{pp} Ramp to $V_{pPL}(1)$

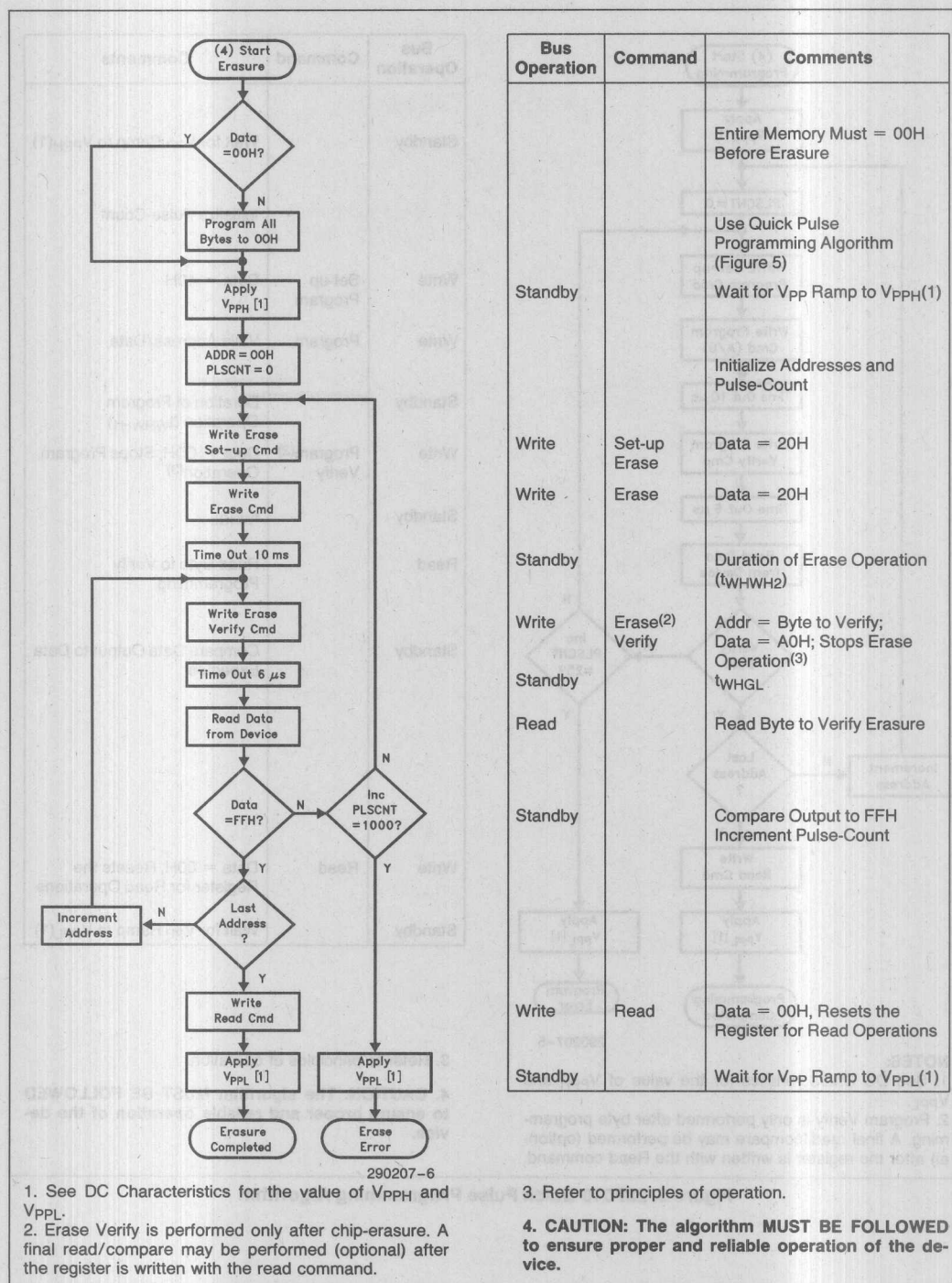


Figure 6. 28F010 Quick Erase Algorithm

DESIGN CONSIDERATIONS

Two-Line Output Control

Flash-memories are often used in larger memory arrays. Intel provides two read-control inputs to accommodate multiple memory connections. Two-line control provides for:

- the lowest possible memory power dissipation and,
- complete assurance that output bus contention will not occur.

To efficiently use these two control inputs, an address-decoder output should drive chip-enable, while the system's read signal controls all flash-memories and other parallel memories. This assures that only enabled memory devices have active outputs, while deselected devices maintain the low power standby condition.

Power Supply Decoupling

Flash-memory power-switching characteristics require careful device decoupling. System designers are interested in three supply current (I_{CC}) issues—standby, active, and transient current peaks produced by falling and rising edges of chip-enable. The capacitive and inductive loads on the device outputs determine the magnitudes of these peaks.

Two-line control and proper decoupling capacitor selection will suppress transient voltage peaks. Each device should have a 0.1 μ F ceramic capacitor connected between V_{CC} and V_{SS} , and between V_{PP} and V_{SS} .

Place the high-frequency, low-inherent-inductance capacitors as close as possible to the devices. Also, for every eight devices, a 4.7 μ F electrolytic capacitor should be placed at the array's power supply connection, between V_{CC} and V_{SS} . The bulk capacitor will overcome voltage slumps caused by printed-

circuit-board trace inductance, and will supply charge to the smaller capacitors as needed.

V_{PP} Trace on Printed Circuit Boards

Programming flash-memories, while they reside in the target system, requires that the printed circuit board designer pay attention to the V_{PP} power supply trace. The V_{PP} pin supplies the memory cell current for programming. Use similar trace widths and layout considerations given the V_{CC} power bus. Adequate V_{PP} supply traces and decoupling will decrease V_{PP} voltage spikes and overshoots.

Power Up/Down Protection

The 28F010 is designed to offer protection against accidental erasure or programming during power transitions. Upon power-up, the 28F010 is indifferent as to which power supply, V_{PP} or V_{CC} , powers up first. Power supply sequencing is not required. Internal circuitry in the 28F010 ensures that the command register is reset to the read mode on power up.

A system designer must guard against active writes for V_{CC} voltages above V_{LKO} when V_{PP} is active. Since both $WE\#$ and $CE\#$ must be low for a command write, driving either to V_{IH} will inhibit writes. The control register architecture provides an added level of protection since alteration of memory contents only occurs after successful completion of the two-step command sequences.

28F010 Power Dissipation

When designing portable systems, designers must consider battery power consumption not only during device operation, but also for data retention during system idle time. Flash nonvolatility increases the usable battery life of your system because the 28F010 does not consume any power to retain code or data when the system is off. Table 4 illustrates the power dissipated when updating the 28F010.

Table 4. 28F010 Typical Update Power Dissipation(4)

Operation	Notes	Power Dissipation (Watt-Seconds)
Array Program/Program Verify	1	0.171
Array Erase/Erase Verify	2	0.136
One Complete Cycle	3	0.478

NOTES:

- Formula to calculate typical Program/Program Verify Power = $[V_{PP} \times \# \text{ Bytes} \times \text{typical} \# \text{ Prog Pulses} (t_{WHWH1} \times I_{PP2} \text{ typical} + t_{WHGL} \times I_{PP4} \text{ typical})] + [V_{CC} \times \# \text{ Bytes} \times \text{typical} \# \text{ Prog Pulses} (t_{WHWH1} \times I_{CC2} \text{ typical} + t_{WHGL} \times I_{CC4} \text{ typical})]$.
- Formula to calculate typical Erase/Erase Verify Power = $[V_{PP} (V_{PP3} \text{ typical} \times t_{ERASE} \text{ typical} + I_{PP5} \text{ typical} \times t_{WHGL} \times \# \text{ Bytes})] + [V_{CC} (I_{CC3} \text{ typical} \times t_{ERASE} \text{ typical} + I_{CC5} \text{ typical} \times t_{WHGL} \times \# \text{ Bytes})]$.
- One Complete Cycle = Array Preprogram + Array Erase + Program.
- "Typicals" are not guaranteed, but based on a limited number of samples from production lots.

ABSOLUTE MAXIMUM RATINGS*

Operating Temperature	
During Read0°C to +70°C(1)
During Erase/Program0°C to +70°C(1)
Operating Temperature	
During Read-40°C to +85°C(2)
During Erase/Program-40°C to +85°C(2)
Temperature Under Bias-10°C to +80°C(1)
Temperature Under Bias-50°C to +95°C(2)
Storage Temperature-65°C to +125°C
Voltage on Any Pin with	
Respect to Ground-2.0V to +7.0V(3)
Voltage on Pin A ₉ with	
Respect to Ground-2.0V to +13.5V(3, 4)
V _{PP} Supply Voltage with	
Respect to Ground	
During Erase/Program-2.0V to +14.0V(3, 4)
V _{CC} Supply Voltage with	
Respect to Ground-2.0V to +7.0V(3)
Output Short Circuit Current100 mA(5)

NOTES:

1. Operating Temperature is for commercial product as defined by this specification.
2. Operating Temperature is for extended temperature products as defined by this specification.
3. Minimum DC input voltage is -0.5V. During transitions, inputs may undershoot to -2.0V for periods less than 20 ns. Maximum DC voltage on output pins is V_{CC} + 0.5V, which may overshoot to V_{CC} + 2.0V for periods less than 20 ns.
4. Maximum DC voltage on A₉ or V_{PP} may overshoot to +14.0V for periods less than 20 ns.
5. Output shorted for no more than one second. No more than one output shorted at a time.
6. See High Speed AC Input/Output reference Waveforms and High Speed AC Testing Load Circuits for testing characteristics.
7. See AC Input/Output reference Waveforms and AC Testing Load Circuits for testing characteristics.

OPERATING CONDITIONS

Symbol	Parameter	Limits		Unit
		Min	Max	
T _A	Operating Temperature	0	70	°C
T _A	Operating Temperature	-40	+85	°C
V _{CC}	V _{CC} Supply Voltage (10%)	4.50	5.50	V
V _{CC}	V _{CC} Supply Voltage (5%)	4.75	5.25	V

DC CHARACTERISTICS—TTL/NMOS COMPATIBLE—Commercial Products

Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typical(4)	Max		
I _{LI}	Input Leakage Current	1			±1.0	μA	V _{CC} = V _{CC} Max V _{IN} = V _{CC} or V _{SS}
I _{LO}	Output Leakage Current	1			±10	μA	V _{CC} = V _{CC} Max V _{OUT} = V _{CC} or V _{SS}
I _{CCS}	V _{CC} Standby Current	1		0.3	1.0	mA	V _{CC} = V _{CC} Max CE# = V _{IH}
I _{CC1}	V _{CC} Active Read Current	1		10	30	mA	V _{CC} = V _{CC} Max, CE# = V _{IL} f = 6 MHz, I _{OUT} = 0 mA

NOTICE: This is a production data sheet. The specifications are subject to change without notice.

***WARNING:** Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.

DC CHARACTERISTICS—TTL/NMOS COMPATIBLE—Commercial Products

(Continued)

Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typical(4)	Max		
I _{CC2}	V _{CC} Programming Current	1, 2		1.0	10	mA	Programming in Progress
I _{CC3}	V _{CC} Erase Current	1, 2		5.0	15	mA	Erasure in Progress
I _{CC4}	V _{CC} Program Verify Current	1, 2		5.0	15	mA	V _{pp} = V _{ppH} , Program Verify in Progress
I _{CC5}	V _{CC} Erase Verify Current	1, 2		5.0	15	mA	V _{pp} = V _{ppH} , Erase Verify in Progress
I _{PPS}	V _{pp} Leakage Current	1			± 10	μA	V _{pp} ≤ V _{CC}
I _{PP1}	V _{pp} Read Current or Standby Current	1		90	200	μA	V _{pp} > V _{CC}
					± 10.0	μA	V _{pp} ≤ V _{CC}
I _{PP2}	V _{pp} Programming Current	1, 2		8.0	30	mA	V _{pp} = V _{ppH} , Programming in Progress
I _{PP3}	V _{pp} Erase Current	1, 2		6.0	30	mA	V _{pp} = V _{ppH} , Erasure in Progress
I _{PP4}	V _{pp} Program Verify Current	1, 2		2.0	5.0	mA	V _{pp} = V _{ppH} , Program Verify in Progress
I _{PP5}	V _{pp} Erase Verify Current	1, 2		2.0	5.0	mA	V _{pp} = V _{ppH} , Erase Verify in Progress
V _{IL}	Input Low Voltage		− 0.5		0.8	V	
V _{IH}	Input High Voltage		2.0		V _{CC} + 0.5	V	
V _{OL}	Output Low Voltage				0.45	V	I _{OL} = 5.8 mA V _{CC} = V _{CC} Min
V _{OH1}	Output High Voltage		2.4			V	I _{OH} = − 2.5 mA V _{CC} = V _{CC} Min
V _{ID}	A ₉ Intelligent Identifier Voltage		11.50		13.00	V	
I _{ID}	A ₉ Intelligent Identifier Current	1, 2		90	200	μA	A ₉ = V _{ID}
V _{PP1}	V _{pp} during Read-Only Operations		0.00		6.5	V	NOTE: Erase/Program are Inhibited when V _{pp} = V _{PP1}
V _{PPH}	V _{pp} during Read/Write Operations		11.40		12.60	V	
V _{LKO}	V _{CC} Erase/Write Lock Voltage		2.5			V	

DC CHARACTERISTICS—CMOS COMPATIBLE—Commercial Products

Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typical(4)	Max		
I _{LI}	Input Leakage Current	1			± 1.0	μA	V _{CC} = V _{CC} Max V _{IN} = V _{CC} or V _{SS}
I _{LO}	Output Leakage Current	1			± 10	μA	V _{CC} = V _{CC} Max V _{OUT} = V _{CC} or V _{SS}
I _{CCS}	V _{CC} Standby Current	1		50	100	μA	V _{CC} = V _{CC} Max CE# = V _{CC} ± 0.2V
I _{CC1}	V _{CC} Active Read Current	1		10	30	mA	V _{CC} = V _{CC} Max, CE# = V _{IL} f = 6 MHz, I _{OUT} = 0 mA

DC CHARACTERISTICS—CMOS COMPATIBLE—Commercial Products (Continued)

Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typical(4)	Max		
I _{CC2}	V _{CC} Programming Current	1, 2		1.0	10	mA	Programming in Progress
I _{CC3}	V _{CC} Erase Current	1, 2		5.0	15	mA	Erase in Progress
I _{CC4}	V _{CC} Program Verify Current	1, 2		5.0	15	mA	V _{PP} = V _{PPH} , Program Verify in Progress
I _{CC5}	V _{CC} Erase Verify Current	1, 2		5.0	15	mA	V _{PP} = V _{PPH} , Erase Verify in Progress
I _{PPS}	V _{PP} Leakage Current	1			±10	μA	V _{PP} ≤ V _{CC}
I _{PP1}	V _{PP} Read Current, I _D Current or Standby Current	1		90	200	μA	V _{PP} > V _{CC}
					±10	μA	V _{PP} ≤ V _{CC}
I _{PP2}	V _{PP} Programming Current	1, 2		8.0	30	mA	V _{PP} = V _{PPH} , Programming in Progress
I _{PP3}	V _{PP} Erase Current	1, 2		6.0	30	mA	V _{PP} = V _{PPH} , Erase in Progress
I _{PP4}	V _{PP} Program Verify Current	1, 2		2.0	5.0	mA	V _{PP} = V _{PPH} , Program Verify in Progress
I _{PP5}	V _{PP} Erase Verify Current	1, 2		2.0	5.0	mA	V _{PP} = V _{PPH} , Erase Verify in Progress
V _{IL}	Input Low Voltage		-0.5		0.8	V	
V _{IH}	Input High Voltage		0.7 V _{CC}		V _{CC} + 0.5	V	
V _{OL}	Output Low Voltage				0.45	V	I _{OL} = 5.8 mA V _{CC} = V _{CC} Min
V _{OH1}	Output High Voltage		0.85 V _{CC}			V	I _{OH} = -2.5 mA, V _{CC} = V _{CC} Min
V _{OH2}			V _{CC} - 0.4			V	I _{OH} = -100 μA, V _{CC} = V _{CC} Min
V _{ID}	A ₉ Intelligent Identifier Voltage		11.50		13.00	V	
I _{ID}	A ₉ Intelligent Identifier Current	1, 2		90	200	μA	A ₉ = V _{ID}
V _{PPL}	V _{PP} during Read-Only Operations		0.00		6.5	V	NOTE: Erase/Programs are Inhibited when V _{PP} = V _{PPL}
V _{PPH}	V _{PP} during Read/Write Operations		11.40		12.60	V	
V _{LKO}	V _{CC} Erase/Write Lock Voltage		2.5			V	

DC CHARACTERISTICS—TTL/NMOS COMPATIBLE—Extended Temperature Products

Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typical(4)	Max		
I_{LI}	Input Leakage Current	1			± 1.0	μA	$V_{CC} = V_{CC} \text{ Max}$ $V_{IN} = V_{CC} \text{ or } V_{SS}$
I_{LO}	Output Leakage Current	1			± 10	μA	$V_{CC} = V_{CC} \text{ Max}$ $V_{OUT} = V_{CC} \text{ or } V_{SS}$
I_{CCS}	V_{CC} Standby Current	1		0.3	1.0	mA	$V_{CC} = V_{CC} \text{ Max}$ $CE\# = V_{IH}$
I_{CC1}	V_{CC} Active Read Current	1		10	30	mA	$V_{CC} = V_{CC} \text{ Max}$, $CE\# = V_{IL}$ $f = 6 \text{ MHz}$, $I_{OUT} = 0 \text{ mA}$
I_{CC2}	V_{CC} Programming Current	1, 2		1.0	30	mA	Programming in Progress
I_{CC3}	V_{CC} Erase Current	1, 2		5.0	30	mA	Erase in Progress
I_{CC4}	V_{CC} Program Verify Current	1, 2		5.0	30	mA	$V_{PP} = V_{PPH}$, Program Verify in Progress
I_{CC5}	V_{CC} Erase Verify Current	1, 2		5.0	30	mA	$V_{PP} = V_{PPH}$, Erase Verify in Progress
I_{PPS}	V_{PP} Leakage Current	1			± 10	μA	$V_{PP} \leq V_{CC}$
I_{PP1}	V_{PP} Read Current or Standby Current	1		90	200	μA	$V_{PP} > V_{CC}$
					± 10.0	μA	$V_{PP} \leq V_{CC}$
I_{PP2}	V_{PP} Programming Current	1, 2		8.0	30	mA	$V_{PP} = V_{PPH}$, Programming in Progress
I_{PP3}	V_{PP} Erase Current	1, 2		6.0	30	mA	$V_{PP} = V_{PPH}$, Erase in Progress
I_{PP4}	V_{PP} Program Verify Current	1, 2		2.0	5.0	mA	$V_{PP} = V_{PPH}$, Program Verify in Progress
I_{PP5}	V_{PP} Erase Verify Current	1, 2		2.0	5.0	mA	$V_{PP} = V_{PPH}$, Erase Verify in Progress
V_{IL}	Input Low Voltage		-0.5		0.8	V	
V_{IH}	Input High Voltage		2.0		$V_{CC} + 0.5$	V	
V_{OL}	Output Low Voltage				0.45	V	$I_{OL} = 5.8 \text{ mA}$ $V_{CC} = V_{CC} \text{ Min}$
V_{OH1}	Output High Voltage		2.4			V	$I_{OH} = -2.5 \text{ mA}$ $V_{CC} = V_{CC} \text{ Min}$
V_{ID}	A_9 Intelligent Identifier Voltage		11.50		13.00	V	
I_{ID}	A_9 Intelligent Identifier Current	1, 2		90	500	μA	$A_9 = V_{ID}$
V_{PPL}	V_{PP} during Read-Only Operations		0.00		6.5	V	NOTE: Erase/Program are Inhibited when $V_{PP} = V_{PPL}$
V_{PPH}	V_{PP} during Read/Write Operations		11.40		12.60	V	
V_{LKO}	V_{CC} Erase/Write Lock Voltage		2.5			V	

Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typical(4)	Max		
I_{LI}	Input Leakage Current	1			± 1.0	μA	$V_{CC} = V_{CC} \text{ Max}$ $V_{IN} = V_{CC} \text{ or } V_{SS}$
I_{LO}	Output Leakage Current	1			± 10	μA	$V_{CC} = V_{CC} \text{ Max}$ $V_{OUT} = V_{CC} \text{ or } V_{SS}$
I_{CCS}	V_{CC} Standby Current	1		50	100	μA	$V_{CC} = V_{CC} \text{ Max}$ $CE\# = V_{CC} \pm 0.2V$
I_{CC1}	V_{CC} Active Read Current	1		10	30	mA	$V_{CC} = V_{CC} \text{ Max}$, $CE\# = V_{IL}$ $f = 10 \text{ MHz}$, $I_{OUT} = 0 \text{ mA}$
I_{CC2}	V_{CC} Programming Current	1, 2		1.0	10	mA	Programming in Progress
I_{CC3}	V_{CC} Erase Current	1, 2		5.0	30	mA	Erase in Progress
I_{CC4}	V_{CC} Program Verify Current	1, 2		5.0	30	mA	$V_{PP} = V_{PPH}$, Program Verify in Progress
I_{CC5}	V_{CC} Erase Verify Current	1, 2		5.0	30	mA	$V_{PP} = V_{PPH}$, Erase Verify in Progress
I_{PPS}	V_{PP} Leakage Current	1			± 10	μA	$V_{PP} \leq V_{CC}$
I_{PP1}	V_{PP} Read Current, ID Current or Standby Current	1		90	200	μA	$V_{PP} > V_{CC}$
					± 10	μA	$V_{PP} \leq V_{CC}$
I_{PP2}	V_{PP} Programming Current	1, 2		8.0	30	mA	$V_{PP} = V_{PPH}$ Programming in Progress
I_{PP3}	V_{PP} Erase Current	1, 2		6.0	30	mA	$V_{PP} = V_{PPH}$ Erase in Progress
I_{PP4}	V_{PP} Program Verify Current	1, 2		2.0	5.0	mA	$V_{PP} = V_{PPH}$, Program Verify in Progress
I_{PP5}	V_{PP} Erase Verify Current	1, 2		2.0	5.0	mA	$V_{PP} = V_{PPH}$, Erase Verify in Progress
V_{IL}	Input Low Voltage		-0.5		0.8	V	
V_{IH}	Input High Voltage		$0.7 V_{CC}$		$V_{CC} + 0.5$	V	
V_{OL}	Output Low Voltage				0.45	V	$I_{OL} = 5.8 \text{ mA}$ $V_{CC} = V_{CC} \text{ Min}$
V_{OH1}	Output High Voltage		$0.85 V_{CC}$			V	$I_{OH} = -2.5 \text{ mA}$, $V_{CC} = V_{CC} \text{ Min}$
V_{OH2}			$V_{CC} - 0.4$				$I_{OH} = -100 \mu A$, $V_{CC} = V_{CC} \text{ Min}$
V_{ID}	A_9 Intelligent Identifier Voltage		11.50		13.00	V	
I_{ID}	A_9 Intelligent Identifier Current	1, 2		90	500	μA	$A_9 = V_{ID}$

DC CHARACTERISTICS—CMOS COMPATIBLE—Extended Temperature Products (Continued)

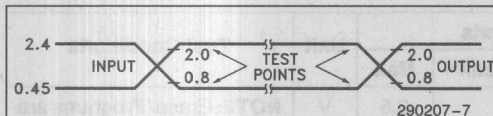
Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typical(4)	Max		
V _{PPL}	V _{PP} during Read-Only Operations		0.00		6.5	V	NOTE: Erase/Programs are Inhibited when V _{PP} = V _{PPL}
V _{PPH}	V _{PP} during Read/Write Operations		11.40		12.60	V	
V _{LKO}	V _{CC} Erase/Write Lock Voltage		2.5			V	

CAPACITANCE T_A = 25°C, f = 1.0 MHz

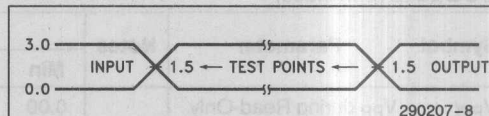
Symbol	Parameter	Notes	Limits		Unit	Conditions
			Min	Max		
C _{IN}	Address/Control Capacitance	3		8	pF	V _{IN} = 0V
C _{OUT}	Output Capacitance	3		12	pF	V _{OUT} = 0V

NOTES:

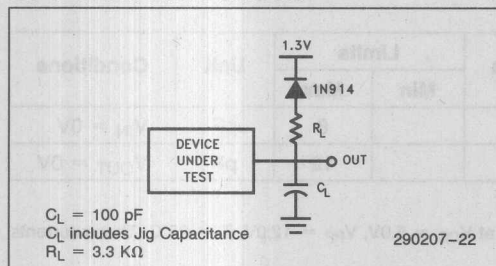
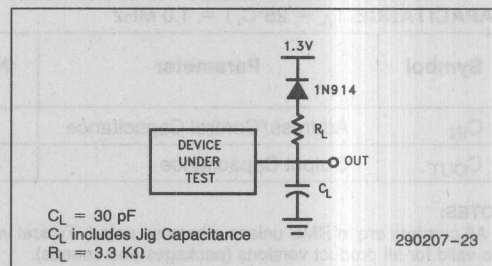
1. All currents are in RMS unless otherwise noted. Typical values at V_{CC} = 5.0V, V_{PP} = 12.0V, T = 25°C. These currents are valid for all product versions (packages and speeds).
2. Not 100% tested: characterization data available.
3. Sampled, not 100% tested.
4. "Typicals" are not guaranteed, but based on a limited number of samples from production lots.

AC TESTING INPUT/OUTPUT WAVEFORM(1)

AC test inputs are driven at V_{OH} (2.4 V_{TTL}) for a Logic "1" and V_{OL} (0.45 V_{TTL}) for a Logic "0". Input timing begins at V_{IH} (2.0 V_{TTL}) and V_{IL} (0.8 V_{TTL}). Output timing ends at V_{IH} and V_{IL} . Input rise and fall times (10% to 90%) < 10 ns.

HIGH SPEED AC TESTING INPUT/OUTPUT WAVEFORM(2)

AC test inputs are driven at 3.0V for a Logic "1" and 0.0V for a Logic "0". Input timing begins, and output timing ends, at 1.5V. Input rise and fall times (10% to 90%) < 10 ns.

AC TESTING LOAD CIRCUIT(1)**HIGH SPEED AC TESTING LOAD CIRCUIT(2)****AC TEST CONDITIONS(1)**

Input Rise and Fall Times (10% to 90%) 10 ns
Input Pulse Levels 0.45V and 2.4V
Input Timing Reference Level 0.8V and 2.0V
Output Timing Reference Level 0.8V and 2.0V
Capacitive Load 100 pF

HIGH-SPEED AC TEST CONDITIONS(2)

Input Rise and Fall Times (10% to 90%) 10 ns
Input Pulse Levels 0.0V and 3.0V
Input Timing Reference Level 1.5V
Output Timing Reference Level 1.5V
Capacitive Load 30 pF

NOTES:

1. Testing characteristics for 28F010-65 in standard configuration, and 28F010-90, 28F010-120, and 28F010-150.
2. Testing characteristics for 28F010-65 in high speed configuration.

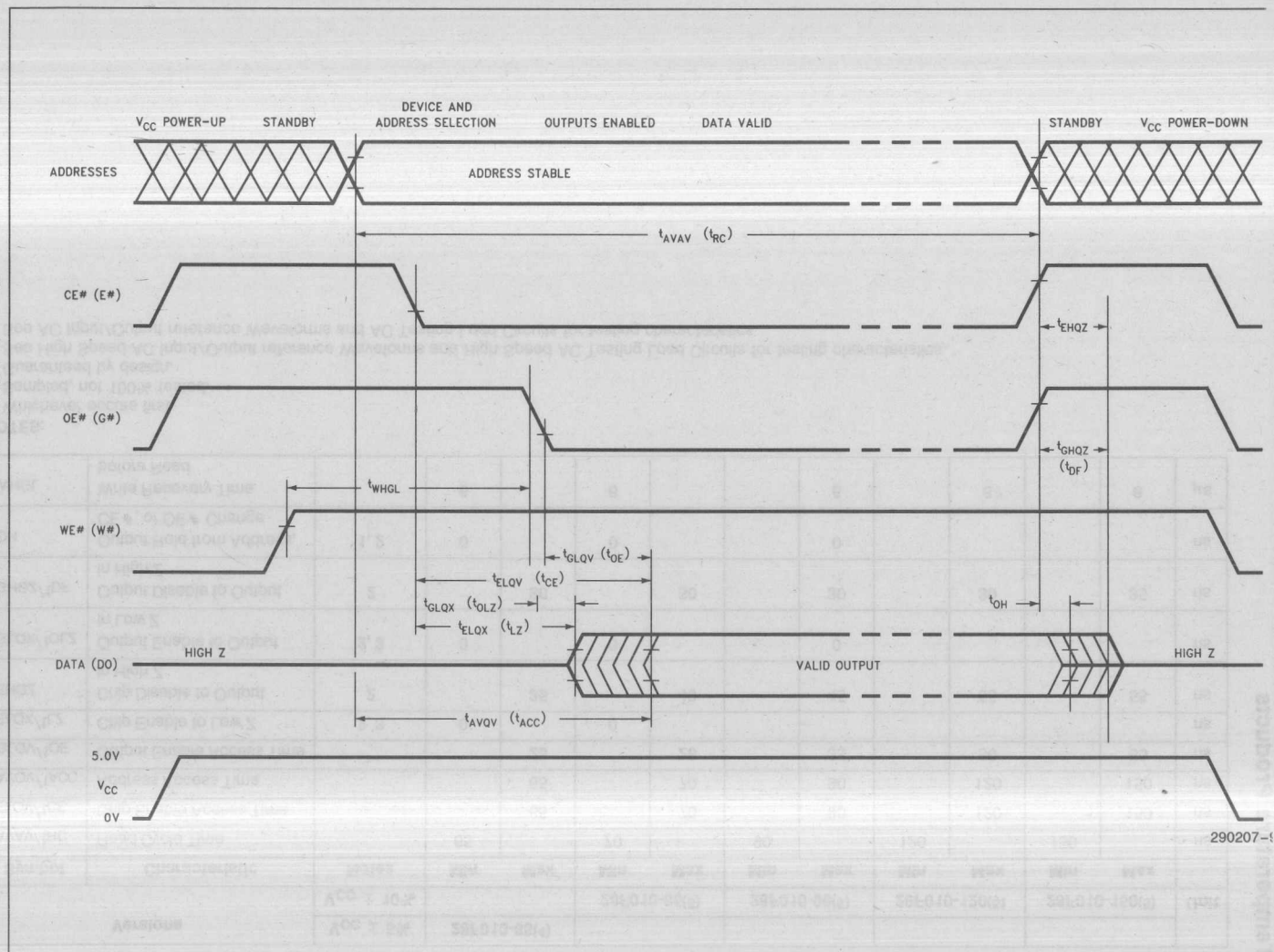
AC CHARACTERISTICS—Read Only Operations—Commercial and Extended Temperature Products

Versions		V _{CC} ± 5%	28F010-65(4)										Unit
		V _{CC} ± 10%			28F010-65(5)		28F010-90(5)		28F010-120(5)		28F010-150(5)		
Symbol	Characteristic	Notes	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	
t _{AVAV} /t _{RC}	Read Cycle Time		65		70		90		120		150		ns
t _{ELQV} /t _{CE}	Chip Enable Access Time			65		70		90		120		150	ns
t _{AVQV} /t _{ACC}	Address Access Time			65		70		90		120		150	ns
t _{GLQV} /t _{OE}	Output Enable Access Time			25		28		35		50		55	ns
t _{ELQX} /t _{LZ}	Chip Enable to Low Z	2, 3	0		0								ns
t _{EHQZ}	Chip Disable to Output in High Z	2		35		40		45		55		55	ns
t _{GLQX} /t _{OLZ}	Output Enable to Output in Low Z	2, 3	0		0			0					ns
t _{GHQZ} /t _{DF}	Output Disable to Output in High Z	2		30		30		30		30		35	ns
t _{OH}	Output Hold from Address, CE #, or OE # Change	1, 2	0		0			0					ns
t _{WHGL}	Write Recovery Time before Read		6		6			6		6		6	μs

NOTES:

1. Whichever occurs first.
2. Sampled, not 100% tested.
3. Guaranteed by design.
4. See High Speed AC Input/Output reference Waveforms and High Speed AC Testing Load Circuits for testing characteristics.
5. See AC Input/Output reference Waveforms and AC Testing Load Circuits for testing characteristics.

Figure 7. AC Waveforms for Read Operations



AC CHARACTERISTICS—Write/Erase/Program Only Operations(1)—
Commercial and Extended Temperature Products

Versions		V _{CC} ± 5%	28F010-65(4)										Unit
		V _{CC} ± 10%			28F010-65(5)		28F010-90(5)		28F010-120(5)		28F010-150(5)		
Symbol	Characteristic	Notes	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	
t _{AVAV} /t _{WC}	Write Cycle Time		65		70		90		120		150		ns
t _{AVWL} /t _{AS}	Address Set-Up Time		0		0		0		0		0		ns
t _{WLAX} /t _{AH}	Address Hold Time		40		40		40		40		40		ns
		6				55							
t _{DVWH} /t _{DS}	Data Set-Up Time		40		40		40		40		40		ns
		6				55							
t _{WHDX} /t _{DH}	Data Hold Time		10		10		10		10		10		ns
t _{WHGL}	Write Recovery Time before Read		6		6		6		6		6		μs
t _{GHWL}	Read Recovery Time before Write	2	0		0		0		0		0		ns
t _{ELWL} /t _{CS}	Chip Enable Set-Up Time before Write		15		15		15		15		15		ns
t _{WHEH} /t _{CH}	Chip Enable Hold Time		0		0		0		0		0		ns
t _{WLWH} /t _{WP}	Write Pulse Width		40		40		40		60		60		ns
		6				55							
t _{WHWL} /t _{WPH}	Write Pulse Width High		20		20		20		20		20		ns
t _{WHWH1}	Duration of Programming Operation	3	10		10		10		10		10		μs
t _{WHWH2}	Duration of Erase Operation	3	9.5		9.5		9.5		9.5		9.5		ms
t _{VPEL}	V _{PP} Set-Up Time to Chip Enable Low	2	1		1		1		1		1		μs

NOTES:

1. Read timing characteristics during read/write operations are the same as during read-only operations. Refer to AC Characteristics for Read-Only Operations.
2. Guaranteed by design.
3. The integrated stop timer terminates the programming/erase operations, thus eliminating the need for a maximum specification.
4. See High Speed AC Input/Output reference Waveforms and High Speed AC Testing Load Circuits for testing characteristics.
5. See AC Input/Output reference Waveforms and AC Testing Load Circuits for testing characteristics.
6. Minimum specification for Extended Temperature product.

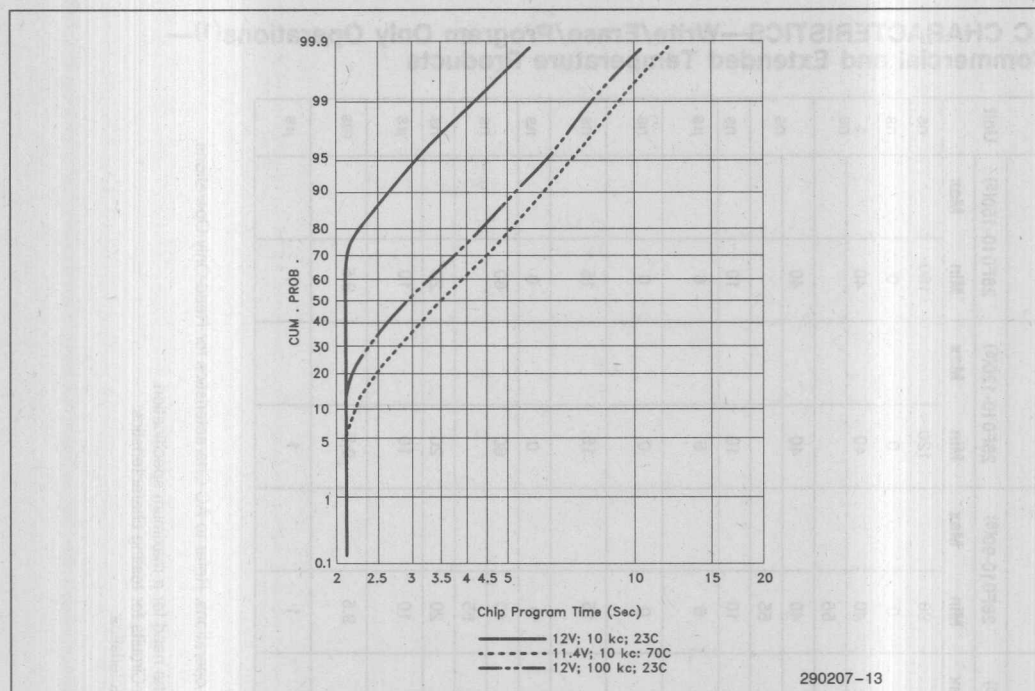


Figure 8. Typical Programming Capability

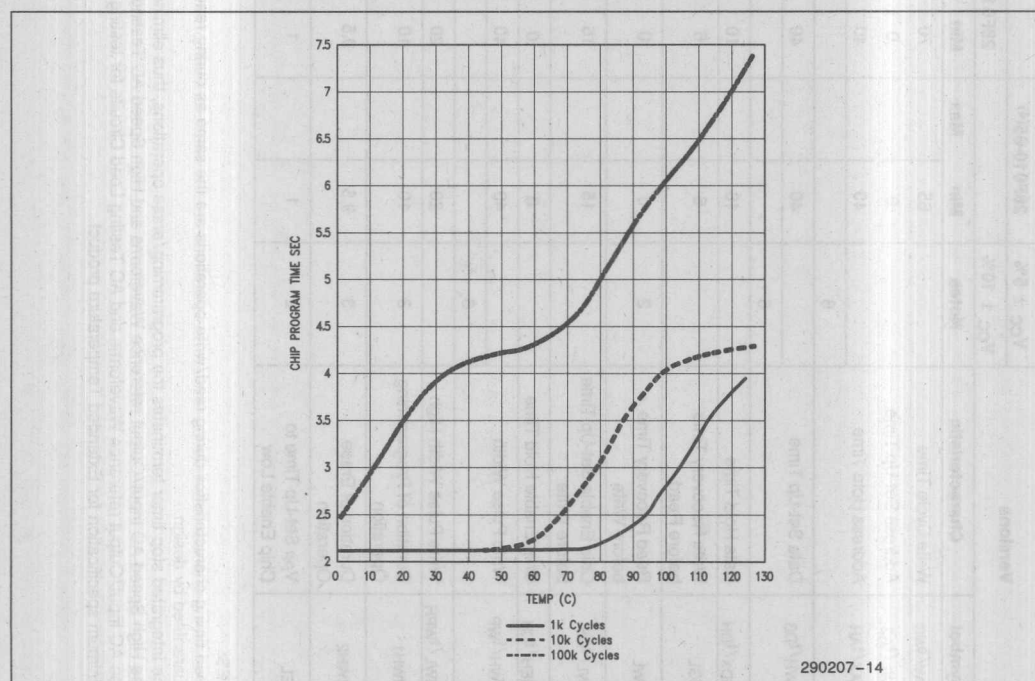
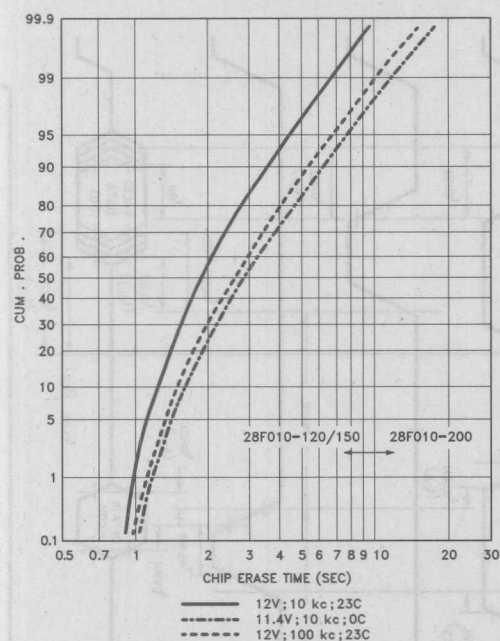
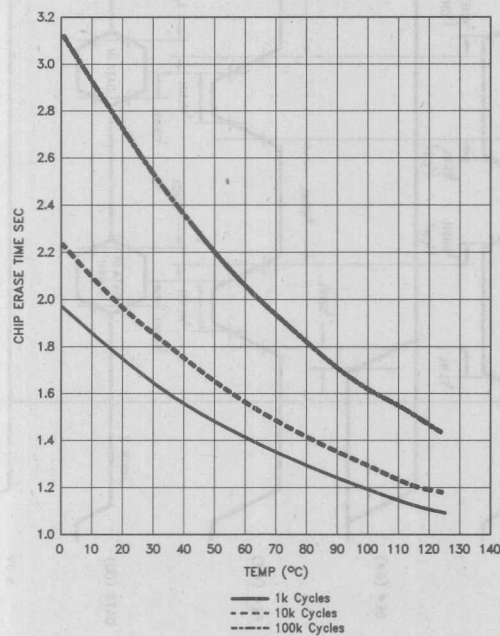


Figure 9. Typical Program Time at 12V



290207-15

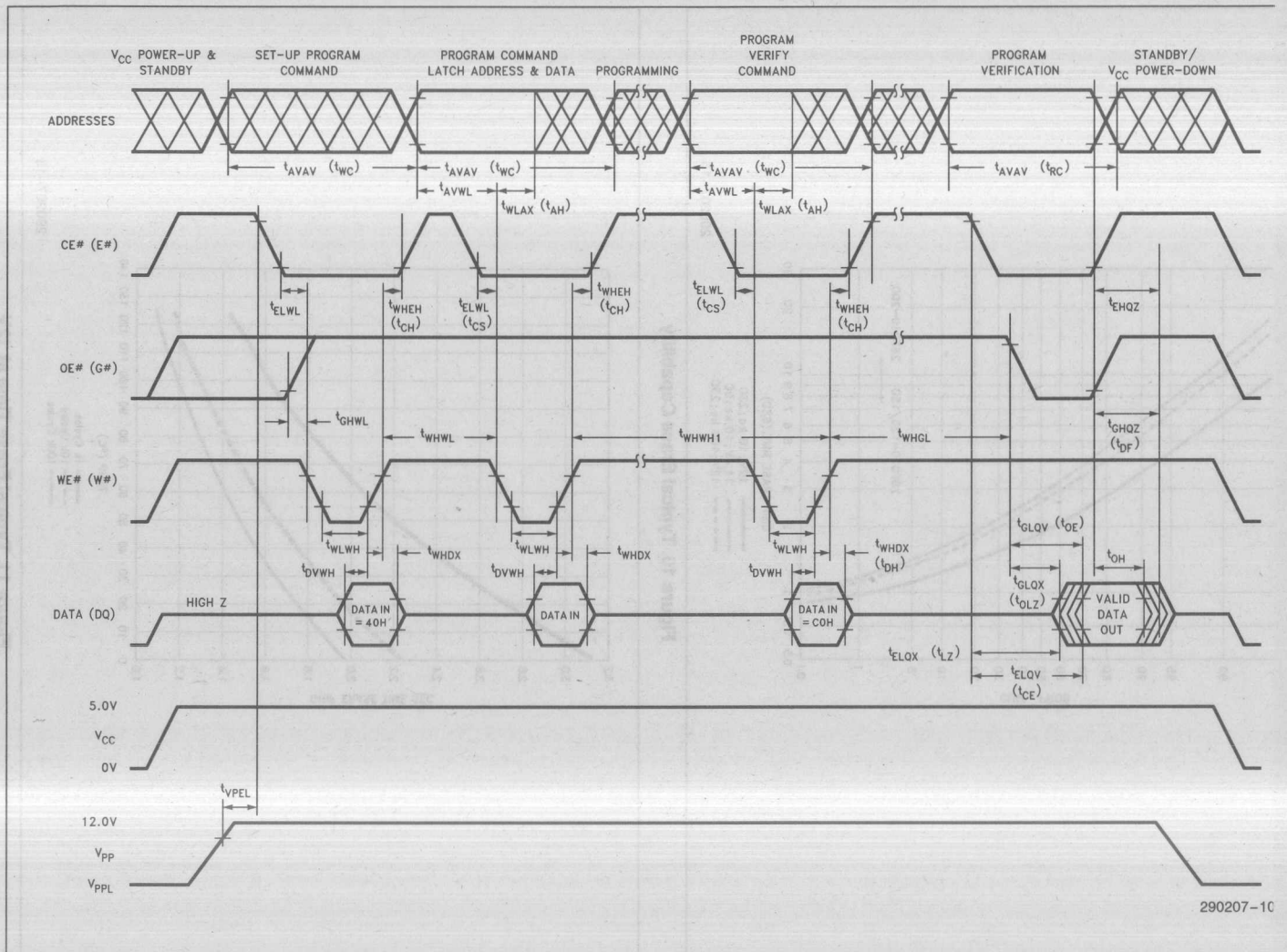
Figure 10. Typical Erase Capability



290207-16

Figure 11. Typical Erase Time at 12V

Figure 12. AC Waveforms for Programming Operations



5-59



AC CHARACTERISTICS—Alternative CE #—Controlled Writes—Commercial and Extended Temperature

Versions		V _{CC} ± 5%	28F010-65(2, 4)										Unit
		V _{CC} ± 10%			28F010-65 ⁽⁵⁾		28F010-90 ⁽⁵⁾		28F010-120 ⁽⁵⁾		28F010-150 ⁽⁵⁾		
Symbol	Characteristic	Notes	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	
t _{AVAV}	Write Cycle Time		65		70		90		120		150		ns
t _{AVEL}	Address Set-Up Time		0		0		0		0		0		ns
t _{ELAX}	Address Hold Time		45		45		45		55		55		ns
		6				60							ns
t _{DVEH}	Data Set-Up Time		35		35		35		45		45		ns
		6				50							ns
t _{EHDx}	Data Hold Time		10		10		10		10		10		ns
t _{EHGL}	Write Recovery Time before Read		6		6		6		6		6		μs
t _{GHWL}	Read Recovery Time before Write	2	0		0		0		0		0		ns
t _{WLEL}	Write Enable Set-Up Time before Chip Enable		0		0		0		0		0		ns
t _{EHWH}	Write Enable Hold Time		0		0		0		0		0		ns
t _{ELEH}	Write Pulse Width		45		45		45		70		70		ns
		6				60							ns
t _{EHEL}	Write Pulse Width High		20		20		20		20		20		ns
t _{EHEH1}	Duration of Programming Operation	3	10		10		10		10		10		μs
t _{EHEH2}	Duration of Erase Operation	3	9.5		9.5		9.5		9.5		9.5		ms
t _{YPEL}	V _{PP} Set-Up Time to Chip Enable Low	2	1		1		1		1		1		μs

NOTE:

1. Read timing characteristics during read/write operations are the same as during read-only operations. Refer to AC Characteristics for Read-Only Operations.
2. Guaranteed by design.
3. The integrated stop timer terminates the programming/erase operations, thus eliminating the need for a maximum specification.
4. See High Speed AC Input/Output reference Waveforms and High Speed AC Testing Load Circuits for testing characteristics.
5. See AC Input/Output reference Waveforms and AC Testing Load Circuits for testing characteristics.
6. Minimum specification for Extended Temperature product.

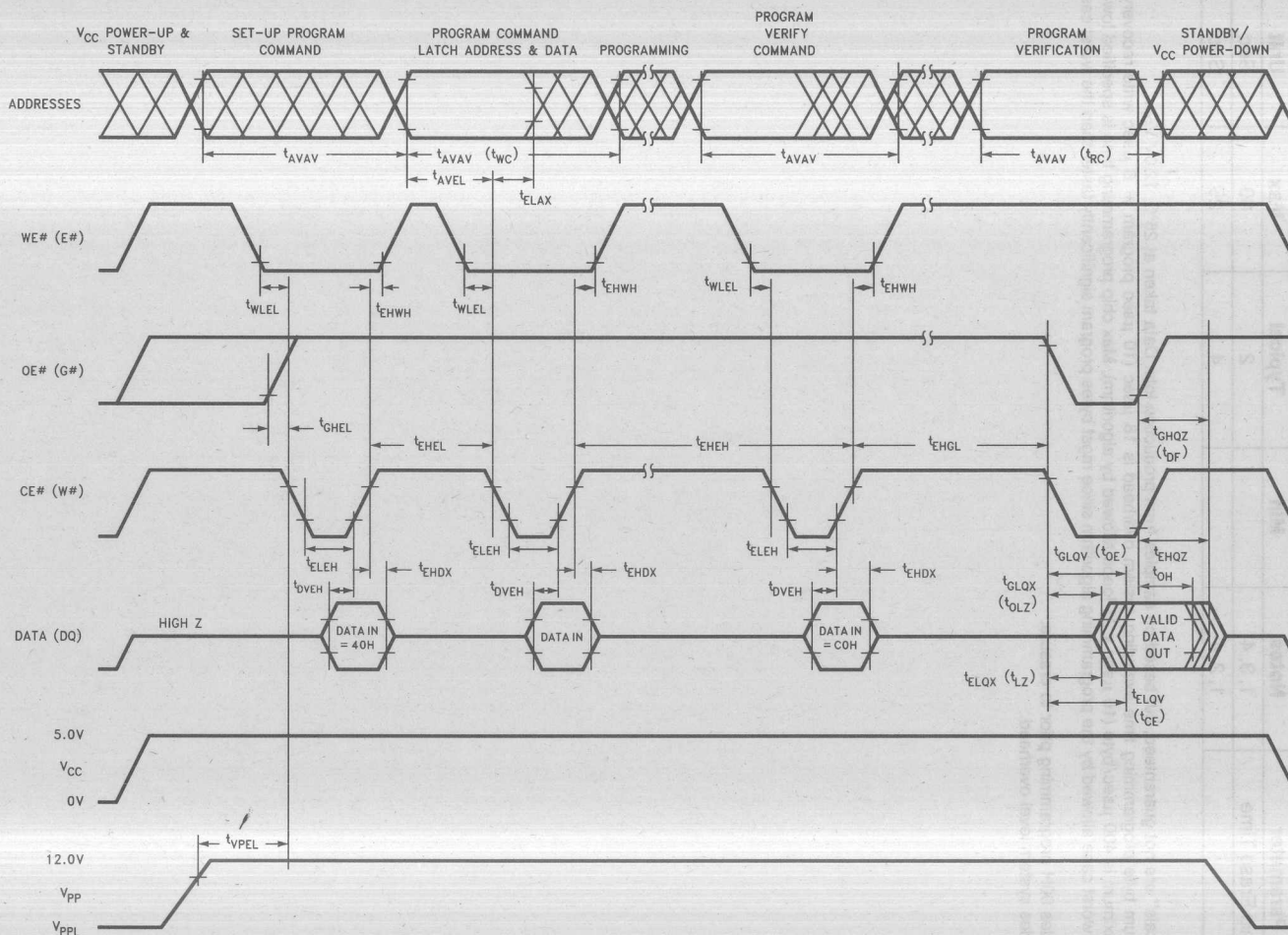
ERASE AND PROGRAMMING PERFORMANCE

Parameter	Notes	Min	Typical	Max	Unit
Chip Erase Time	1, 3, 4		2	30	Sec
	1, 2, 4		4	25	Sec

NOTES:

1. "Typicals" are not guaranteed, but based on samples from production lots. Data taken at 25°C, 12.0V V_{pp}.
2. Minimum byte programming time excluding system overhead is 16 μsec (10 μsec program + 6 μsec write recovery), while maximum is 400 μsec/byte (16 μsec x 25 loops allowed by algorithm). Max chip programming time is specified lower than the worst case allowed by the programming algorithm since most bytes program significantly faster than the worst case byte.
3. Excludes 00H programming prior to erasure.
4. Excludes system level overhead.

Figure 14. Alternate AC Waveforms for Programming Operations



ORDERING INFORMATION

T	P	2	8	F	0	1	0	-	1	2	0
---	---	---	---	---	---	---	---	---	---	---	---

TEMPERATURE ———— PACKAGE ———— ACCESS SPEED (ns)

T = EXTENDED (-40°C to +85°C) P = 32-PIN PLASTIC DIP 120 ns
BLANK = COMMERCIAL (0°C to +70°C) N = 32-LEAD PLCC 150 ns
E = STANDARD 32-LEAD TSOP
F = REVERSE 32-LEAD TSOP

290207-20

VALID COMBINATIONS:

P28F010-65	N28F010-65	TN28F010-90
P28F010-90	N28F010-90	
P28F010-120	N28F010-120	
P28F010-150	N28F010-150	
E28F010-65	F28F010-65	TE28F010-90
E28F010-90	F28F010-90	TF28F010-90
E28F010-120	F28F010-120	
E28F010-150	F28F010-150	

ADDITIONAL INFORMATION

		Order Number
ER-20,	"ETOX Flash Memory Technology"	294005
ER-24,	"Intel Flash Memory"	294008
ER-28,	"ETOX III Flash Memory Technology"	294012
RR-60,	"ETOX Flash Memory Reliability Data Summary"	293002
AP-316,	"Using Flash Memory for In-System Reprogrammable Nonvolatile Storage"	292046
AP-325	"Guide to Flash Memory Reprogramming"	292059

REVISION HISTORY

Number	Description
007	Removed 200 ns Speed Bin Revised Erase Maximum Pulse Count for Figure 5 from 3000 to 1000 Clarified AC and DC Test Conditions Added "dimple" to F TSOP Package Corrected Serpentine Layout
008	Corrected AC Waveforms Added Extended Temperature Options
009	Added 28F010-65 and 28F010-90 speeds Revised Symbols, i.e., CE, OE, etc. to CE#, OE#, etc.

- (See Packaging Spec., Order #231369)

66

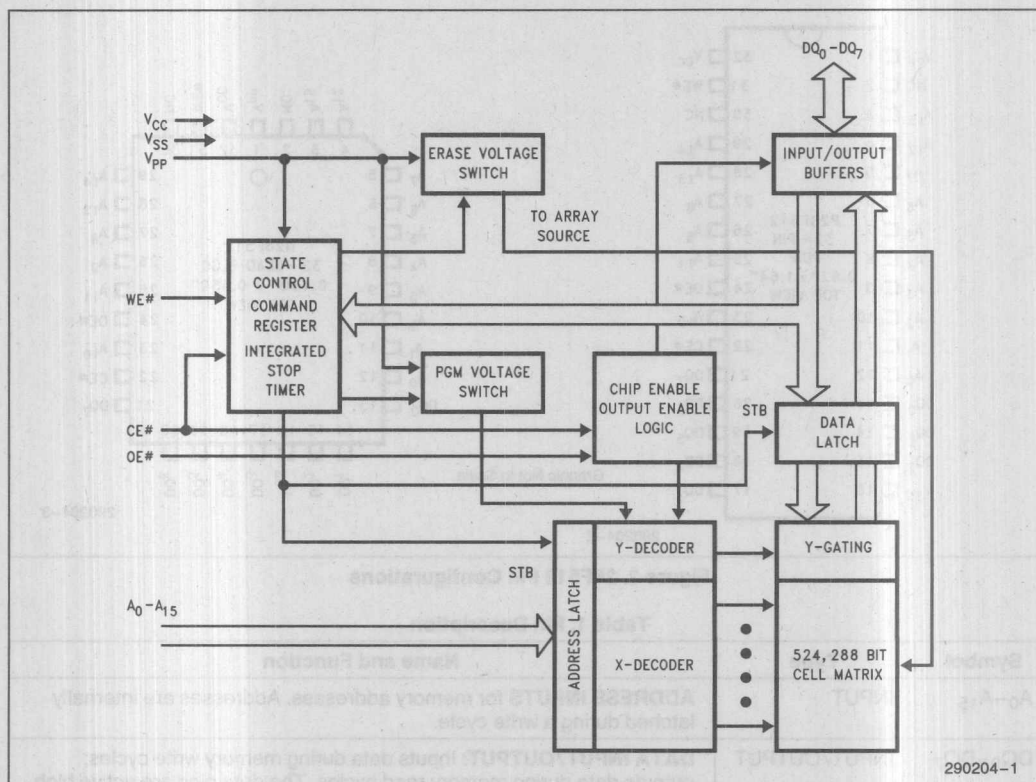


Figure 1. 28F512 Block Diagram

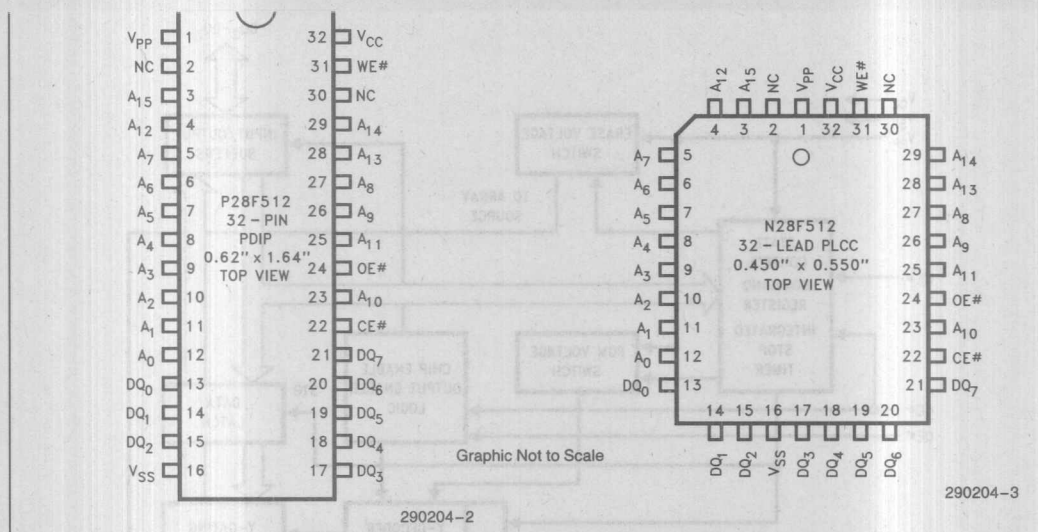


Figure 2. 28F512 Pin Configurations

Table 1. Pin Description

Symbol	Type	Name and Function
A ₀ -A ₁₅	INPUT	ADDRESS INPUTS for memory addresses. Addresses are internally latched during a write cycle.
DQ ₀ -DQ ₇	INPUT/OUTPUT	DATA INPUT/OUTPUT: Inputs data during memory write cycles; outputs data during memory read cycles. The data pins are active high and float to tri-state OFF when the chip is deselected or the outputs are disabled. Data is internally latched during a write cycle.
CE #	INPUT	CHIP ENABLE: Activates the device's control logic, input buffers, decoders and sense amplifiers. CE # is active low; CE # high deselects the memory device and reduces power consumption to standby levels.
OE #	INPUT	OUTPUT ENABLE: Gates the devices output through the data buffers during a read cycle. OE # is active low.
WE #	INPUT	WRITE ENABLE: Controls writes to the control register and the array. Write enable is active low. Addresses are latched on the falling edge and data is latched on the rising edge of the WE # pulse. Note: With V _{pp} ≤ 6.5V, memory contents cannot be altered.
V _{pp}		ERASE/PROGRAM POWER SUPPLY for writing the command register, erasing the entire array, or programming bytes in the array.
V _{CC}		DEVICE POWER SUPPLY (5V ± 10%)
V _{SS}		GROUND
NC		NO INTERNAL CONNECTION to device. Pin may be driven or left floating.

APPLICATIONS

The 28F512 flash memory provides nonvolatility along with the capability to perform over 100,000 electrical chip-erase/reprogram cycles. These features make the 28F512 an innovative alternative to disk, EEPROM, and battery-backed static RAM. Where periodic updates of code and data-tables are required, the 28F512's reprogrammability and nonvolatility make it the obvious and ideal replacement for EPROM.

Primary applications and operating systems stored in flash eliminate the slow disk-to-DRAM download process. This results in dramatic enhancement of performance and substantial reduction of power consumption — a consideration particularly important in portable equipment. Flash memory increases flexibility with electrical chip erasure and in-system update capability of operating systems and application code. With updatable BIOS, system manufacturers can easily accommodate last-minute changes as revisions are made.

In diskless workstations and terminals, network traffic reduces to a minimum and systems are instant-on. Reliability exceeds that of electromechanical media. Often in these environments, power interruptions force extended re-boot periods for all networked terminals. This mishap is no longer an issue if boot code, operating systems, communication protocols and primary applications are flash-resident in each terminal.

For embedded systems that rely on dynamic RAM/disk for main system memory or nonvolatile backup storage, the 28F512 flash memory offers a solid state alternative in a minimal form factor. The 28F512 provides higher performance, lower power consumption, instant-on capability, and allows an "execute in place" memory hierarchy for code and data table reading. Additionally, the flash memory is more rugged and reliable in harsh environments where extreme temperatures and shock can cause disk-based systems to fail.

The need for code updates pervades all phases of a system's life — from prototyping to system manufacture to after-sale service. The electrical chip-erase and reprogramming ability of the 28F512 allows in-

circuit alterability; this eliminates unnecessary handling and less-reliable socketed connections, while adding greater test, manufacture, and update flexibility.

Material and labor costs associated with code changes increases at higher levels of system integration — the most costly being code updates after sale. Code "bugs", or the desire to augment system functionality, prompt after-sale code updates. Field revisions to EPROM-based code requires the removal of EPROM components or entire boards. With the 28F512, code updates are implemented locally via an edge-connector, or remotely over a communication link.

For systems currently using a high-density static RAM/battery configuration for data accumulation, flash memory's inherent nonvolatility eliminates the need for battery backup. The concern for battery failure no longer exists, an important consideration for portable equipment and medical instruments, both requiring continuous performance. In addition, flash memory offers a considerable cost advantage over static RAM.

Flash memory's electrical chip erasure, byte programmability and complete nonvolatility fit well with data accumulation and recording needs. Electrical chip-erase gives the designer a "blank slate" in which to log or record data. Data can be periodically off-loaded for analysis and the flash memory erased producing a new "blank slate".

A high degree of on-chip feature integration simplifies memory-to-processor interfacing. Figure 3 depicts two 28F512s tied to the 80C186 system bus. The 28F512's architecture minimizes interface circuitry needed for complete in-circuit updates of memory contents.

With cost-effective in-system reprogramming, extended cycling capability, and true nonvolatility, the 28F512 offers advantages to the alternatives: EPROMs, EEPROMs, battery backed static RAM, or disk. EPROM-compatible read specifications, straight-forward interfacing, and in-circuit alterability offers designers unlimited flexibility to meet the high standards of today's designs.

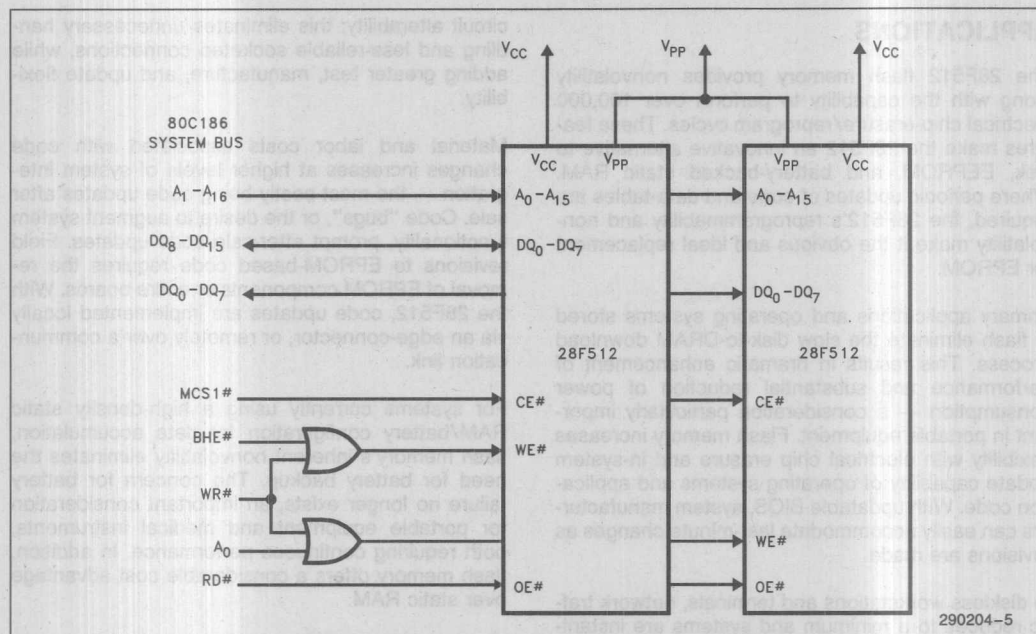


Figure 3. 28F512 in a 80C186 System

PRINCIPLES OF OPERATION

Flash-memory augments EPROM functionality with in-circuit electrical erasure and reprogramming. The 28F512 introduces a command register to manage this new functionality. The command register allows for: 100% TTL-level control inputs; fixed power supplies during erasure and programming; and maximum EPROM compatibility.

In the absence of high voltage on the Vpp pin, the 28F512 is a read-only memory. Manipulation of the external memory-control pins yields the standard EPROM read, standby, output disable, and Intelligent Identifier operations.

The same EPROM read, standby, and output disable operations are available when high voltage is applied to the Vpp pin. In addition, high voltage on Vpp enables erasure and programming of the device. All functions associated with altering memory contents—Intelligent Identifier, erase, erase verify, program, and program verify—are accessed via the command register.

Commands are written to the register using standard microprocessor write timings. Register contents serve as input to an internal state-machine which controls the erase and programming circuitry. Write cycles also internally latch addresses and data needed for programming or erase operations. With the appropriate command written to the register,

standard microprocessor read timings output array data, access the Intelligent Identifier codes, or output data for erase and program verification.

Integrated Stop Timer

Successive command write cycles define the duration of program and erase operations; specifically, the program or erase time durations are normally terminated by associated program or erase verify commands. An integrated stop timer provides simplified timing control over these operations; thus eliminating the need for maximum program/erase timing specifications. Programming and erase pulse durations are minimums only. When the stop timer terminates a program or erase operation, the device enters an inactive state and remains inactive until receiving the appropriate verify or reset command.

Write Protection

The command register is only active when Vpp is at high voltage. Depending upon the application, the system designer may choose to make the Vpp power supply switchable—available only when memory updates are desired. When $V_{pp} = V_{ppL}$, the contents of the register default to the read command, making the 28F512 a read-only memory. In this mode, the memory contents cannot be altered.

Operation		V _{PP} (¹)	A ₀	A ₉	CE #	OE #	WE #	DQ ₀ -DQ ₇
READ-ONLY	Read	V _{PPL}	A ₀	A ₉	V _{IL}	V _{IL}	V _{IH}	Data Out
	Output Disable	V _{PPL}	X	X	V _{IL}	V _{IH}	V _{IH}	Tri-State
	Standby	V _{PPL}	X	X	V _{IH}	X	X	Tri-State
	Intelligent Identifier (Mfr)(²)	V _{PPL}	V _{IL}	V _{ID} (³)	V _{IL}	V _{IL}	V _{IH}	Data = 89H
	Intelligent Identifier (Device)(²)	V _{PPL}	V _{IH}	V _{ID} (³)	V _{IL}	V _{IL}	V _{IH}	Data = B8H
READ/WRITE	Read	V _{PPH}	A ₀	A ₉	V _{IL}	V _{IL}	V _{IH}	Data Out(⁴)
	Output Disable	V _{PPH}	X	X	V _{IL}	V _{IH}	V _{IH}	Tri-State
	Standby(⁵)	V _{PPH}	X	X	V _{IH}	X	X	Tri-State
	Write	V _{PPH}	A ₀	A ₉	V _{IL}	V _{IH}	V _{IL}	Data In(⁶)

NOTES:

1. Refer to DC Characteristics. When V_{PP} = V_{PPL} memory contents can be read but not written or erased.
2. Manufacturer and device codes may also be accessed via a command register write sequence. Refer to Table 3. All other addresses low.
3. V_{ID} is the Intelligent Identifier high voltage. Refer to DC Characteristics.
4. Read operations with V_{PP} = V_{PPH} may access array data or the Intelligent Identifier codes.
5. With V_{PP} at high voltage, the standby current equals I_{CC} + I_{PP} (standby).
6. Refer to Table 3 for valid Data-In during a write operation.
7. X can be V_{IL} or V_{IH}.

Or, the system designer may choose to "hardwire" V_{PP}, making the high voltage supply constantly available. In this case, all Command Register functions are inhibited whenever V_{CC} is below the write lockout voltage V_{LKO}. (See Power Up/Down Protection). The 28F512 is designed to accommodate either design practice, and to encourage optimization of the processor-memory interface.

The two-step program/erase write sequence to the Command Register provides additional software write protection.

BUS OPERATIONS**Read**

The 28F512 has two control functions, both of which must be logically active, to obtain data at the outputs. Chip-Enable (CE #) is the power control and should be used for device selection. Output-Enable (OE #) is the output control and should be used to gate data from the output pins, independent of device selection. Refer to AC read timing waveforms.

When V_{PP} is high (V_{PPH}), the read operation can be used to access array data, to output the Intelligent Identifier codes, and to access data for program/erase verification. When V_{PP} is low (V_{PPL}), the read operation can **only** access the array data.

Output Disable

With Output-Enable at a logic-high level (V_{IH}), output from the device is disabled. Output pins are placed in a high-impedance state.

Standby

With Chip-Enable at a logic-high level, the standby operation disables most of the 28F512's circuitry and substantially reduces device power consumption. The outputs are placed in a high-impedance state, independent of the Output-Enable signal. If the 28F512 is deselected during erasure, programming, or program/erase verification, the device draws active current until the operation is terminated.

Intelligent Identifier Operation

The Intelligent Identifier operation outputs the manufacturer code (89H) and device code (B8H). Programming equipment automatically matches the device with its proper erase and programming algorithms.

With Chip-Enable and Output-Enable at a logic low level, raising A9 to high voltage V_{ID} (see DC Characteristics) activates the operation. Data read from locations 0000H and 0001H represent the manufacturer's code and the device code, respectively.

The manufacturer- and device-codes can also be read via the command register, for instances where the 28F512 is erased and reprogrammed in the target system. Following a write of 90H to the command register, a read from address location 0000H outputs the manufacturer code (89H). A read from address 0001H outputs the device code (B8H).

Write

Device erasure and programming are accomplished via the command register, when high voltage is applied to the V_{pp} pin. The contents of the register serve as input to the internal state-machine. The state-machine outputs dictate the function of the device.

The command register itself does not occupy an addressable memory location. The register is a latch

used to store the command, along with address and data information needed to execute the command.

The command register is written by bringing Write-Enable to a logic-low level (V_{IL}), while Chip-Enable is low. Addresses are latched on the falling edge of Write-Enable, while data is latched on the rising edge of the Write-Enable pulse. Standard microprocessor write timings are used.

Refer to AC Write Characteristics and the Erase/Programming Waveforms for specific timing parameters.

COMMAND DEFINITIONS

When low voltage is applied to the V_{pp} pin, the contents of the command register default to 00H, enabling read-only operations.

Placing high voltage on the V_{pp} pin enables read/write operations. Device operations are selected by writing specific data patterns into the command register. Table 3 defines these 28F512 register commands.

Table 3. Command Definitions

Command	Bus Cycles Req'd	First Bus Cycle			Second Bus Cycle		
		Operation(1)	Address(2)	Data(3)	Operation(1)	Address(2)	Data(3)
Read Memory	1	Write	X	00H			
Read Intelligent Identifier Code(4)	3	Write	X	90H	Read	(4)	(4)
Set-up Erase/Erase(5)	2	Write	X	20H	Write	X	20H
Erase Verify(5)	2	Write	EA	A0H	Read	X	EVD
Set-up Program/Program(6)	2	Write	X	40H	Write	PA	PD
Program Verify(6)	2	Write	X	C0H	Read	X	PVD
Reset(7)	2	Write	X	FFH	Write	X	FFH

NOTES:

- Bus operations are defined in Table 2.
- IA = Identifier address: 00H for manufacturer code, 01H for device code.
EA = Address of memory location to be read during erase verify.
PA = Address of memory location to be programmed.
Addresses are latched on the falling edge of the Write-Enable pulse.
- ID = Data read from location IA during device identification (Mfr = 89H, Device = B8H).
EVD = Data read from location EA during erase verify.
PD = Data to be programmed at location PA. Data is latched on the rising edge of Write-Enable.
PVD = Data read from location PA during program verify. PA is latched on the Program command.
- Following the Read Intelligent ID command, two read operations access manufacturer and device codes.
- Figure 5 illustrates the Quick-Erase algorithm.
- Figure 4 illustrates the Quick-Pulse Programming algorithm.
- The second bus cycle must be followed by the desired command register write.

Read Command

While V_{pp} is high, for erasure and programming, memory contents can be accessed via the read command. The read operation is initiated by writing 00H into the command register. Microprocessor read cycles retrieve array data. The device remains enabled for reads until the command register contents are altered.

The default contents of the register upon V_{pp} power-up is 00H. This default value ensures that no spurious alteration of memory contents occurs during the V_{pp} power transition. Where the V_{pp} supply is hard-wired to the 28F512, the device powers-up and remains enabled for reads until the command-register contents are changed. Refer to the AC Read Characteristics and Waveforms for specific timing parameters.

Intelligent Identifier Command

Flash-memories are intended for use in applications where the local CPU alters memory contents. As such, manufacturer- and device-codes must be accessible while the device resides in the target system. PROM programmers typically access signature codes by raising A9 to a high voltage. However, multiplexing high voltage onto address lines is not a desired system-design practice.

The 28F512 contains an Intelligent Identifier operation to supplement traditional PROM-programming methodology. The operation is initiated by writing 90H into the command register. Following the command write, a read cycle from address 0000H retrieves the manufacturer code of 89H. A read cycle from address 0001H returns the device code of B8H. To terminate the operation, it is necessary to write another valid command into the register.

Set-up Erase/Erase Commands

Set-up Erase is a command-only operation that stages the device for electrical erasure of all bytes in the array. The set-up erase operation is performed by writing 20H to the command register.

To commence chip-erasure, the erase command (20H) must again be written to the register. The erase operation begins with the rising edge of the Write-Enable pulse and terminates with the rising edge of the next Write-Enable pulse (i.e., Erase-Verify Command).

This two-step sequence of set-up followed by execution ensures that memory contents are not accidentally erased. Also, chip-erasure can only occur when high voltage is applied to the V_{pp} pin. In the absence

of this high voltage, memory contents are protected against erasure. Refer to AC Erase Characteristics and Waveforms for specific timing parameters.

Erase-Verify Command

The erase command erases all bytes of the array in parallel. After each erase operation, all bytes must be verified. The erase verify operation is initiated by writing A0H into the command register. The address for the byte to be verified must be supplied as it is latched on the falling edge of the Write-Enable pulse. The register write terminates the erase operation with the rising edge of its Write-Enable pulse.

The 28F512 applies an internally-generated margin voltage to the addressed byte. Reading FFH from the addressed byte indicates that all bits in the byte are erased.

The erase-verify command must be written to the command register prior to each byte verification to latch its address. The process continues for each byte in the array until a byte does not return FFH data, or the last address is accessed.

In the case where the data read is not FFH, another erase operation is performed. (Refer to Set-up Erase/Erasure). Verification then resumes from the address of the last-verified byte. Once all bytes in the array have been verified, the erase step is complete. The device can be programmed. At this point, the verify operation is terminated by writing a valid command (e.g. Program Set-up) to the command register. Figure 5, the Quick-Erase algorithm, illustrates how commands and bus operations are combined to perform electrical erasure of the 28F512. Refer to AC Erase Characteristics and Waveforms for specific timing parameters.

5

Set-up Program/Program Commands

Set-up program is a command-only operation that stages the device for byte programming. Writing 40H into the command register performs the set-up operation.

Once the program set-up operation is performed, the next Write-Enable pulse causes a transition to an active programming operation. Addresses are internally latched on the falling edge of the Write-Enable pulse. Data is internally latched on the rising edge of the Write-Enable pulse. The rising edge of Write-Enable also begins the programming operation. The programming operation terminates with the next rising edge of Write-Enable, used to write the program-verify command. Refer to AC Programming Characteristics and Waveforms for specific timing parameters.

Program-Verify Command

The 28F512 is programmed on a byte-by-byte basis. Byte programming may occur sequentially or at random. Following each programming operation, the byte just programmed must be verified.

The program-verify operation is initiated by writing C0H into the command register. The register write terminates the programming operation with the rising edge of its Write-Enable pulse. The program-verify operation stages the device for verification of the byte last programmed. No new address information is latched.

The 28F512 applies an internally-generated margin voltage to the byte. A microprocessor read cycle outputs the data. A successful comparison between the programmed byte and true data means that the byte is successfully programmed. Programming then proceeds to the next desired byte location. Figure 4, the 28F512 Quick-Pulse Programming algorithm, illustrates how commands are combined with bus operations to perform byte programming. Refer to AC Programming Characteristics and Waveforms for specific timing parameters.

Reset Command

A reset command is provided as a means to safely abort the erase- or program-command sequences. Following either set-up command (erase or program) with two consecutive writes of FFH will safely abort the operation. Memory contents will not be altered. A valid command must then be written to place the device in the desired state.

EXTENDED ERASE/PROGRAM CYCLING

EEPROM cycling failures have always concerned users. The high electrical field required by thin oxide EEPROMs for tunneling can literally tear apart the oxide at defect regions. To combat this, some suppliers have implemented redundancy schemes, reducing cycling failures to insignificant levels. However, redundancy requires that cell size be doubled—an expensive solution.

Intel has designed extended cycling capability into its ETOX II flash memory technology. Resulting improvements in cycling reliability come without increasing memory cell size or complexity. First, an advanced tunnel oxide increases the charge carrying ability ten-fold. Second, the oxide area per cell subjected to the tunneling electric field is one-tenth that of common EEPROMs, minimizing the probability of oxide defects in the region. Finally, the peak electric field during erasure is approximately 2 MV/cm lower than EEPROM. The lower electric field

greatly reduces oxide stress and the probability of failure—increasing time to wearout by a factor of 100,000,000.

The 28F512 is capable of 100,000 program/erase cycles. The device is programmed and erased using Intel's Quick-Pulse Programming and Quick-Erase algorithms. Intel's algorithmic approach uses a series of operations (pulses), along with byte verification, to completely and reliably erase and program the device.

For further information, see Reliability Report RR-60 (ETOX-II Reliability Data Summary).

QUICK-PULSE PROGRAMMING ALGORITHM

The Quick-Pulse Programming algorithm uses programming operations of 10 μ s duration. Each operation is followed by a byte verification to determine when the addressed byte has been successfully programmed. The algorithm allows for up to 25 programming operations per byte, although most bytes verify on the first or second operation. The entire sequence of programming and byte verification is performed with V_{PP} at high voltage. Figure 4 illustrates the Quick-Pulse Programming algorithm.

QUICK-ERASE ALGORITHM

Intel's Quick-Erase algorithm yields fast and reliable electrical erasure of memory contents. The algorithm employs a closed-loop flow, similar to the Quick-Pulse Programming algorithm, to simultaneously remove charge from all bits in the array.

Erase begins with a read of memory contents. The 28F512 is erased when shipped from the factory. Reading FFH data from the device would immediately be followed by device programming.

For devices being erased and reprogrammed, uniform and reliable erasure is ensured by first programming all bits in the device to their charged state (Data = 00H). This is accomplished, using the Quick-Pulse Programming algorithm, in approximately one second.

Erase execution then continues with an initial erase operation. Erase verification (data = FFH) begins at address 0000H and continues through the array to the last address, or until data other than FFH is encountered. With each erase operation, an increasing number of bytes verify to the erased state. Erase efficiency may be improved by storing the address of the last byte verified in a register. Following the next erase operation, verification starts at that stored address location. Erasure typically occurs in one second. Figure 5 illustrates the Quick-Erase algorithm.

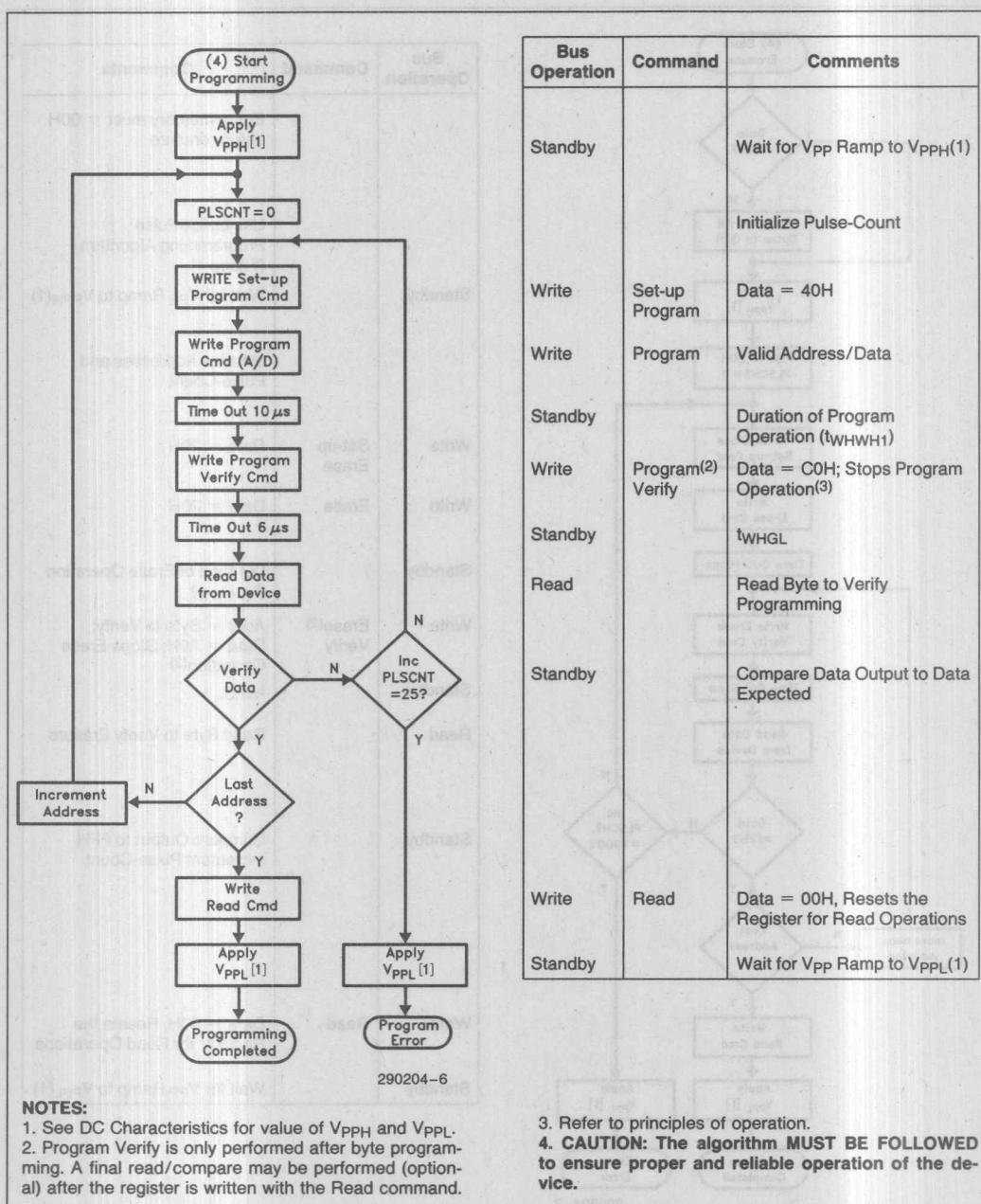
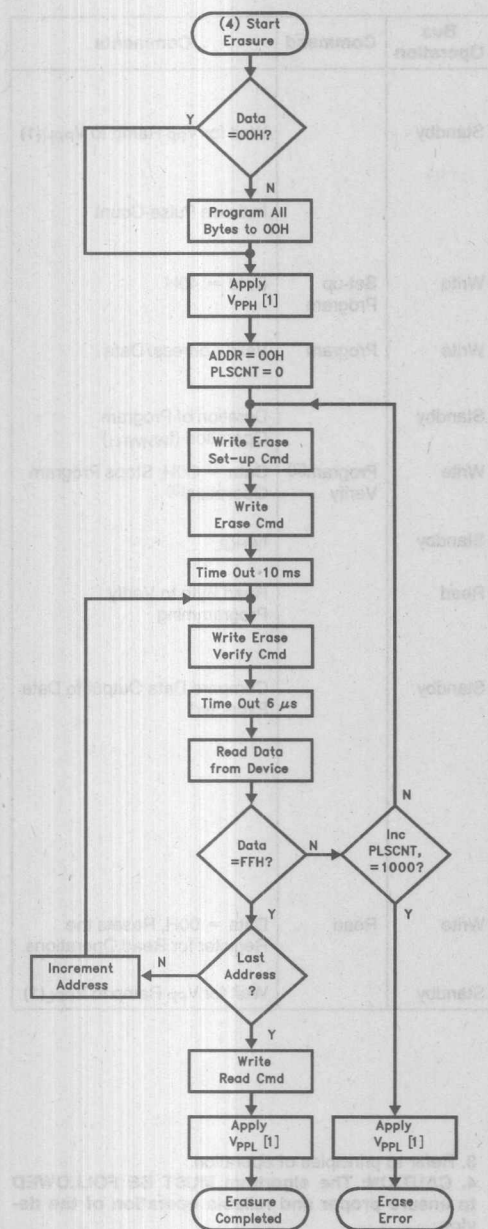


Figure 4. 28F512 Quick-Pulse Programming Algorithm



290204-7

NOTES:

1. See DC Characteristics for value of V_{ppH} and V_{pPL} .
2. Erase Verify is performed only after chip-erasure. A final read/compare may be performed (optional) after the register is written with the read command.

Bus Operation	Command	Comments
		Entire memory must = 00H before erasure
Standby		Use Quick-Pulse Programming Algorithm (Figure 4) Wait for V_{pp} Ramp to $V_{ppH}(1)$
Write	Set-up Erase	Data = 20H
Write	Erase	Data = 20H
Standby		Duration of Erase Operation (t_{WHWH2})
Write	Erase(2) Verify	Addr = Byte to Verify; Data = A0H; Stops Erase Operation(3)
Standby		t_{WHGL}
Read		Read Byte to Verify Erasure
Standby		Compare Output to FFH Increment Pulse-Count
Write	Read	Data = 00H, Resets the Register for Read Operations
Standby		Wait for V_{pp} Ramp to $V_{pPL}(1)$

3. Refer to principles of operation.

4. CAUTION: The algorithm MUST BE FOLLOWED to ensure proper and reliable operation of the device.

Figure 5. 28F512 Quick-Erase Algorithm

DESIGN CONSIDERATIONS

Two-Line Output Control

Flash-memories are often used in larger memory arrays. Intel provides two read-control inputs to accommodate multiple memory connections. Two-line control provides for:

- the lowest possible memory power dissipation and,
- complete assurance that output bus contention will not occur.

To efficiently use these two control inputs, an address-decoder output should drive chip-enable, while the system's read signal controls all flash-memories and other parallel memories. This assures that only enabled memory devices have active outputs, while deselected devices maintain the low power standby condition.

Power Supply Decoupling

Flash-memory power-switching characteristics require careful device decoupling. System designers are interested in three supply current (I_{CC}) issues—standby, active, and transient current peaks produced by falling and rising edges of chip-enable. The capacitive and inductive loads on the device outputs determine the magnitudes of these peaks.

Two-line control and proper decoupling capacitor selection will suppress transient voltage peaks. Each device should have a 0.1 μ F ceramic capacitor connected between V_{CC} and V_{SS} , and between V_{PP} and V_{SS} .

Place the high-frequency, low-inherent-inductance capacitors as close as possible to the devices. Also, for every eight devices, a 4.7 μ F electrolytic capacitor should be placed at the array's power supply connection, between V_{CC} and V_{SS} . The bulk capacitor will overcome voltage slumps caused by printed-circuit-board trace inductance, and will supply charge to the smaller capacitors as needed.

V_{PP} Trace on Printed Circuit Boards

Programming flash-memories, while they reside in the target system, requires that the printed circuit board designer pay attention to the V_{PP} power supply trace. The V_{PP} pin supplies the memory cell current for programming. Use similar trace widths and layout considerations given the V_{CC} power bus. Adequate V_{PP} supply traces and decoupling will decrease V_{PP} voltage spikes and overshoots.

Power Up/Down Protection

The 28F512 is designed to offer protection against accidental erasure or programming during power transitions. Upon power-up, the 28F512 is indifferent as to which power supply, V_{PP} or V_{CC} , powers up first. Power supply sequencing is not required. Internal circuitry in the 28F512 ensures that the command register is reset to the read mode on power up.

A system designer must guard against active writes for V_{CC} voltages above V_{LKO} when V_{PP} is active. Since both $WE\#$ and $CE\#$ must be low for a command write, driving either to V_{IH} will inhibit writes. The control register architecture provides an added level of protection since alteration of memory contents only occurs after successful completion of the two-step command sequences.

28F512 Power Dissipation

When designing portable systems, designers must consider battery power consumption not only during device operation, but also for data retention during system idle time. Flash memory nonvolatility increases the usable battery life of your system because the 28F512 does not consume any power to retain code or data when the system is off. Table 4 illustrates the power dissipated when updating the 28F512.

Table 4. 28F512 Typical Update Power Dissipation⁽⁴⁾

Operation	Notes	Power Dissipation (Watt-Seconds)
Array Program/ Program Verify	1	0.085
Array Erase/ Erase Verify	2	0.092
One Complete Cycle	3	0.262

NOTES:

- Formula to calculate typical Program/Program Verify Power = [$V_{PP} \times \# \text{ Bytes} \times \text{Typical } \# \text{ Prog Pulses} (t_{WHWH1} \times I_{PP2} \text{ Typical} + t_{WHGL} \times I_{PP4} \text{ Typical})$] + [$V_{CC} \times \# \text{ Bytes} \times \text{Typical } \# \text{ Prog Pulses} (t_{WHWH1} \times I_{CC2} \text{ Typical} + t_{WHGL} \times I_{CC4} \text{ Typical})$].
- Formula to calculate typical Erase/Erase Verify Power = [$V_{PP}(I_{PP3} \text{ Typical} \times t_{ERASE} \text{ Typical} + I_{PP5} \text{ Typical} \times t_{WHGL} \times \# \text{ Bytes})$] + [$V_{CC}(I_{CC3} \text{ Typical} \times t_{ERASE} \text{ Typical} + I_{CC5} \text{ Typical} \times t_{WHGL} \times \# \text{ Bytes})$].
- One Complete Cycle = Array Preprogram + Array Erase + Program.
- "Typicals" are not guaranteed, but based on a limited number of samples from production lots.

ABSOLUTE MAXIMUM RATINGS*

Operating Temperature	
During Read	0°C to +70°C(1)
During Erase/Program	0°C to +70°C(1)
Operating Temperature	
During Read	-40°C to +85°C(2)
During Erase/Program	-40°C to +85°C(2)
Temperature Under Bias	
	-10°C to +80°C(1)
Temperature Under Bias	
	-50°C to +95°C(2)
Storage Temperature	
	-65°C to +125°C
Voltage on Any Pin with	
Respect to Ground	-2.0V to +7.0V(2)
Voltage on Pin A _g with	
Respect to Ground	-2.0V to +13.5V(2, 3)

V_{PP} Supply Voltage with
Respect to Ground
During Erase/Program -2.0V to +14.0V(2, 3)

V_{CC} Supply Voltage with
Respect to Ground -2.0V to +7.0V(2)

Output Short Circuit Current 100 mA(4)

NOTICE: This is a production data sheet. The specifications are subject to change without notice.

*WARNING: Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.

NOTES:

1. Operating temperature is for commercial product defined by this specification.
2. Operating temperature is for extended temperature product defined by this specification.
3. Minimum DC input voltage is -0.5V. During transitions, inputs may undershoot to -2.0V for periods less than 20 ns. Maximum DC voltage on output pins is V_{CC} + 0.5V, which may overshoot to V_{CC} + 2.0V for periods less than 20 ns.
4. Maximum DC voltage on A_g or V_{PP} may overshoot to +14.0V for periods less than 20 ns.
5. Output shorted for no more than one second. No more than one output shorted at a time.

OPERATING CONDITIONS

Symbol	Parameter	Limits		Unit	Comments
		Min	Max		
T _A	Operating Temperature(1)	0	70	°C	For Read-Only and Read/Write Operations for Commercial Products
T _A	Operating Temperature(2)	-40	+85	°C	For Read-Only and Read/Write Operations for Extended Temperature Products
V _{CC}	V _{CC} Supply Voltage	4.50	5.50	V	

DC CHARACTERISTICS—TTL/NMOS COMPATIBLE—Commercial Products

Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typ(4)	Max		
I _{LI}	Input Leakage Current	1			±1.0	μA	V _{CC} = V _{CC} Max V _{IN} = V _{CC} or V _{SS}
I _{LO}	Output Leakage Current	1			±10.0	μA	V _{CC} = V _{CC} Max V _{OUT} = V _{CC} or V _{SS}
I _{CCS}	V _{CC} Standby Current	1		0.3	1.0	mA	V _{CC} = V _{CC} Max CE# = V _{IH}
I _{CC1}	V _{CC} Active Read Current	1		10	30	mA	V _{CC} = V _{CC} Max, CE# = V _{IL} f = 6 MHz, I _{OUT} = 0 mA
I _{CC2}	V _{CC} Programming Current	1, 2		1.0	10	mA	Programming in Progress
I _{CC3}	V _{CC} Erase Current	1, 2		5.0	15	mA	Erase in Progress
I _{CC4}	V _{CC} Program Verify Current	1, 2		5.0	15	mA	V _{PP} = V _{PPH} Program Verify in Progress

DC CHARACTERISTICS—TTL/NMOS COMPATIBLE—Commercial Products

(Continued)

Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typ(4)	Max		
I _{CC5}	V _{CC} Erase Verify Current	1, 2		5.0	15	mA	V _{PP} = V _{PPH} Erase Verify in Progress
I _{PPS}	V _{PP} Leakage Current	1			± 10.0	μA	V _{PP} ≤ V _{CC}
I _{PP1}	V _{PP} Read Current, Standby Current, or I _D Current	1		90	200	μA	V _{PP} > V _{CC}
					± 10.0		V _{PP} ≤ V _{CC}
I _{PP2}	V _{PP} Programming Current	1, 2		8.0	30	mA	V _{PP} = V _{PPH} Programming in Progress
I _{PP3}	V _{PP} Erase Current	1, 2		4.0	30	mA	V _{PP} = V _{PPH} Erase in Progress
I _{PP4}	V _{PP} Program Verify Current	1, 2		2.0	5.0	mA	V _{PP} = V _{PPH} Program Verify in Progress
I _{PP5}	V _{PP} Erase Verify Current	1, 2		2.0	5.0	mA	V _{PP} = V _{PPH} Erase Verify in Progress
V _{IL}	Input Low Voltage		−0.5		0.8	V	
V _{IH}	Input High Voltage		2.0		V _{CC} + 0.5	V	
V _{OL}	Output Low Voltage				0.45	V	I _{OL} = 5.8 mA V _{CC} = V _{CC} Min
V _{OH1}	Output High Voltage		2.4			V	I _{OH} = −2.5 mA V _{CC} = V _{CC} Min
V _{ID}	A ₉ Intelligent Identifier Voltage		11.50		13.00	V	
I _{ID}	A ₉ Intelligent Identifier Current	1, 2		90	200	μA	A ₉ = V _{ID}
V _{PPL}	V _{PP} during Read-Only Operations		0.00		6.5	V	NOTE: Erase/Program are inhibited when V _{PP} = V _{PPL}
V _{PPH}	V _{PP} during Read/Write Operations		11.40		12.60	V	
V _{LKO}	V _{CC} Erase/Write Lock Voltage		2.5			V	

DC CHARACTERISTICS—CMOS COMPATIBLE—Commercial Products

Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typ(4)	Max		
I _{LI}	Input Leakage Current	1			± 1.0	μA	V _{CC} = V _{CC} Max V _{IN} = V _{CC} or V _{SS}
I _{LO}	Output Leakage Current	1			± 10.0	μA	V _{CC} = V _{CC} Max V _{OUT} = V _{CC} or V _{SS}
I _{CCS}	V _{CC} Standby Current	1		50	100	μA	V _{CC} = V _{CC} Max CE# = V _{CC} ± 0.2V
I _{CC1}	V _{CC} Active Read Current	1		10	30	mA	V _{CC} = V _{CC} Max, CE# = V _{IL} f = 6 MHz, I _{OUT} = 0 mA

Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typ ⁽⁴⁾	Max		
I _{CC2}	V _{CC} Programming Current	1, 2		1.0	10	mA	Programming in Progress
I _{CC3}	V _{CC} Erase Current	1, 2		5.0	15	mA	Erase in Progress
I _{CC4}	V _{CC} Program Verify Current	1, 2		5.0	15	mA	V _{PP} = V _{PPH} Program Verify in Progress
I _{CC5}	V _{CC} Erase Verify Current	1, 2		5.0	15	mA	V _{PP} = V _{PPH} Erase Verify in Progress
I _{PPS}	V _{PP} Leakage Current	1			± 10.0	μA	V _{PP} ≤ V _{CC}
I _{PP1}	V _{PP} Read Current, I _D Current, or Standby Current	1		90	200	μA	V _{PP} > V _{CC}
					± 10.0		V _{PP} ≤ V _{CC}
I _{PP2}	V _{PP} Programming Current	1, 2		8.0	30	mA	V _{PP} = V _{PPH} Programming in Progress
I _{PP3}	V _{PP} Erase Current	1, 2		4.0	30	mA	V _{PP} = V _{PPH} Erase in Progress
I _{PP4}	V _{PP} Program Verify Current	1, 2		2.0	5.0	mA	V _{PP} = V _{PPH} Program Verify in Progress
I _{PP5}	V _{PP} Erase Verify Current	1, 2		2.0	5.0	mA	V _{PP} = V _{PPH} Erase Verify in Progress
V _{IL}	Input Low Voltage		-0.5		0.8	V	
V _{IH}	Input High Voltage		0.7 V _{CC}		V _{CC} + 0.5	V	
V _{OL}	Output Low Voltage				0.45	V	I _{OL} = 5.8 mA V _{CC} = V _{CC} Min
V _{OH1}	Output High Voltage		0.85 V _{CC}			V	I _{OH} = -2.5 mA, V _{CC} = V _{CC} Min
V _{OH2}			V _{CC} - 0.4				I _{OH} = -100 μA, V _{CC} = V _{CC} Min
V _{ID}	A _g Intelligent Identifier Voltage		11.50		13.00	V	A _g = V _{ID}
I _{ID}	A _g Intelligent Identifier Current	1, 2		90	200	μA	A _g = V _{ID}
V _{PPL}	V _{PP} during Read-Only Operations		0.00		6.5	V	NOTE: Erase/Program are Inhibited when V _{PP} = V _{PPL}
V _{PPH}	V _{PP} during Read/Write Operations		11.40		12.60	V	
V _{LKO}	V _{CC} Erase/Write Lock Voltage		2.5			V	

DC CHARACTERISTICS—TTL/NMOS COMPATIBLE—Extended Temperature Products

Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typ(4)	Max		
I _{LI}	Input Leakage Current	1			± 1.0	μA	V _{CC} = V _{CC} Max V _{IN} = V _{CC} or V _{SS}
I _{LO}	Output Leakage Current	1			± 10.0	μA	V _{CC} = V _{CC} Max V _{OUT} = V _{CC} or V _{SS}
I _{CCS}	V _{CC} Standby Current	1		0.3	1.0	mA	V _{CC} = V _{CC} Max CE# = V _{IH}
I _{CC1}	V _{CC} Active Read Current	1		10	30	mA	V _{CC} = V _{CC} Max, CE# = V _{IL} f = 6 MHz, I _{OUT} = 0 mA
I _{CC2}	V _{CC} Programming Current	1, 2		1.0	30	mA	Programming in Progress
I _{CC3}	V _{CC} Erase Current	1, 2		5.0	30	mA	Erase in Progress
I _{CC4}	V _{CC} Program Verify Current	1, 2		5.0	30	mA	V _{PP} = V _{PPH} Program Verify in Progress
I _{CC5}	V _{CC} Erase Verify Current	1, 2		5.0	30	mA	V _{PP} = V _{PPH} Erase Verify in Progress
I _{PPS}	V _{PP} Leakage Current	1			± 10.0	μA	V _{PP} ≤ V _{CC}
I _{PP1}	V _{PP} Read Current, Standby Current, or ID Current	1		90	200	μA	V _{PP} > V _{CC}
					± 10.0		V _{PP} ≤ V _{CC}
I _{PP2}	V _{PP} Programming Current	1, 2		8.0	30	mA	V _{PP} = V _{PPH} Programming in Progress
I _{PP3}	V _{PP} Erase Current	1, 2		4.0	30	mA	V _{PP} = V _{PPH} Erase in Progress
I _{PP4}	V _{PP} Program Verify Current	1, 2		2.0	5.0	mA	V _{PP} = V _{PPH} Program Verify in Progress
I _{PP5}	V _{PP} Erase Verify Current	1, 2		2.0	5.0	mA	V _{PP} = V _{PPH} Erase Verify in Progress
V _{IL}	Input Low Voltage		−0.5		0.8	V	
V _{IH}	Input High Voltage		2.0		V _{CC} + 0.5	V	
V _{OL}	Output Low Voltage				0.45	V	I _{OL} = 5.8 mA V _{CC} = V _{CC} Min
V _{OH1}	Output High Voltage		2.4			V	I _{OH} = −2.5 mA V _{CC} = V _{CC} Min
V _{ID}	A ₉ Intelligent Identifier Voltage		11.50		13.00	V	
I _{ID}	A ₉ Intelligent Identifier Current	1, 2		90	500	μA	A ₉ = V _{ID}
V _{PPL}	V _{PP} during Read-Only Operations		0.00		6.5	V	NOTE: Erase/Program are Inhibited when V _{PP} = V _{PPL}
V _{PPH}	V _{PP} during Read/Write Operations		11.40		12.60	V	
V _{LKO}	V _{CC} Erase/Write Lock Voltage		2.5			V	

DC CHARACTERISTICS—CMOS COMPATIBLE—Extended Temperature Products

Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typ(4)	Max		
I_{LI}	Input Leakage Current	1			± 1.0	μA	$V_{CC} = V_{CC} \text{ Max}$ $V_{IN} = V_{CC} \text{ or } V_{SS}$
I_{LO}	Output Leakage Current	1			± 10.0	μA	$V_{CC} = V_{CC} \text{ Max}$ $V_{OUT} = V_{CC} \text{ or } V_{SS}$
I_{CCS}	V_{CC} Standby Current	1		50	100	μA	$V_{CC} = V_{CC} \text{ Max}$ $CE\# = V_{CC} \pm 0.2V$
I_{CC1}	V_{CC} Active Read Current	1		10	50	mA	$V_{CC} = V_{CC} \text{ Max}$, $CE\# = V_{IL}$ $f = 6 \text{ MHz}$, $I_{OUT} = 0 \text{ mA}$
I_{CC2}	V_{CC} Programming Current	1, 2		1.0	10	mA	Programming in Progress
I_{CC3}	V_{CC} Erase Current	1, 2		5.0	15	mA	Erase in Progress
I_{CC4}	V_{CC} Program Verify Current	1, 2		5.0	30	mA	$V_{PP} = V_{PPH}$ Program Verify in Progress
I_{CC5}	V_{CC} Erase Verify Current	1, 2		5.0	30	mA	$V_{PP} = V_{PPH}$ Erase Verify in Progress
I_{PPS}	V_{PP} Leakage Current	1			± 10.0	μA	$V_{PP} \leq V_{CC}$
I_{PP1}	V_{PP} Read Current, I_D Current, or Standby Current	1		90	200	μA	$V_{PP} > V_{CC}$
					± 10.0		$V_{PP} \leq V_{CC}$
I_{PP2}	V_{PP} Programming Current	1, 2		8.0	30	mA	$V_{PP} = V_{PPH}$ Programming in Progress
I_{PP3}	V_{PP} Erase Current	1, 2		4.0	30	mA	$V_{PP} = V_{PPH}$ Erase in Progress
I_{PP4}	V_{PP} Program Verify Current	1, 2		2.0	5.0	mA	$V_{PP} = V_{PPH}$ Program Verify in Progress
I_{PP5}	V_{PP} Erase Verify Current	1, 2		2.0	5.0	mA	$V_{PP} = V_{PPH}$ Erase Verify in Progress
V_{IL}	Input Low Voltage		-0.5		0.8	V	
V_{IH}	Input High Voltage		$0.7 V_{CC}$		$V_{CC} + 0.5$	V	
V_{OL}	Output Low Voltage				0.45	V	$I_{OL} = 5.8 \text{ mA}$ $V_{CC} = V_{CC} \text{ Min}$

DC CHARACTERISTICS—CMOS COMPATIBLE—Extended Temperature Products (Continued)

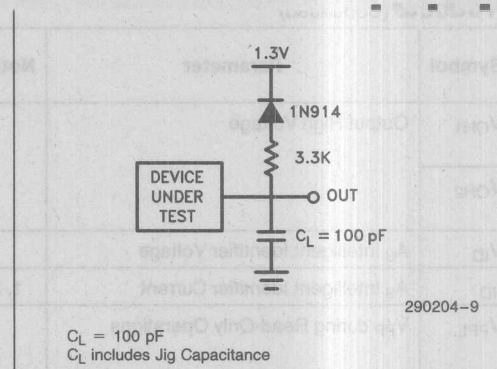
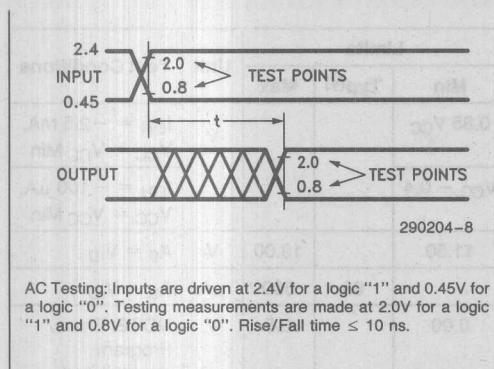
Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typ(4)	Max		
V _{OH1}	Output High Voltage		0.85 V _{CC}			V	I _{OH} = -2.5 mA, V _{CC} = V _{CC} Min
V _{OH2}			V _{CC} - 0.4				I _{OH} = -100 µA, V _{CC} = V _{CC} Min
V _{ID}	A ₉ Intelligent Identifier Voltage		11.50		13.00	V	A ₉ = V _{ID}
I _{ID}	A ₉ Intelligent Identifier Current	1, 2		90	500	µA	A ₉ = V _{ID}
V _{PPL}	V _{PP} during Read-Only Operations		0.00		6.5	V	NOTE: Erase/ Program are Inhibited when V _{PP} = V _{PPL}
V _{PPH}	V _{PP} during Read/Write Operations		11.40		12.60	V	
V _{LKO}	V _{CC} Erase/Write Lock Voltage		2.5			V	

CAPACITANCE T_A = 25°C, f = 1.0 MHz

Symbol	Parameter	Notes	Limits		Unit	Conditions
			Min	Max		
C _{IN}	Address/Control Capacitance	3		8	pF	V _{IN} = 0V
C _{OUT}	Output Capacitance	3		12	pF	V _{OUT} = 0V

NOTES:

1. All currents are in RMS unless otherwise noted. Typical values at V_{CC} = 5.0V, V_{PP} = 12.0V, T = +25°C. These currents are valid for all product versions (packages and speeds).
2. Not 100% tested: characterization data available.
3. Sampled, not 100% tested.
4. "Typicals" are not guaranteed, but based on a limited number of samples from production lots.



AC TEST CONDITIONS

Input Rise and Fall Times (10% to 90%) 10 ns
 Input Pulse Levels 0.45V and 2.4V
 Input Timing Reference Level 0.8V and 2.0V
 Output Timing Reference Level 0.8V and 2.0V

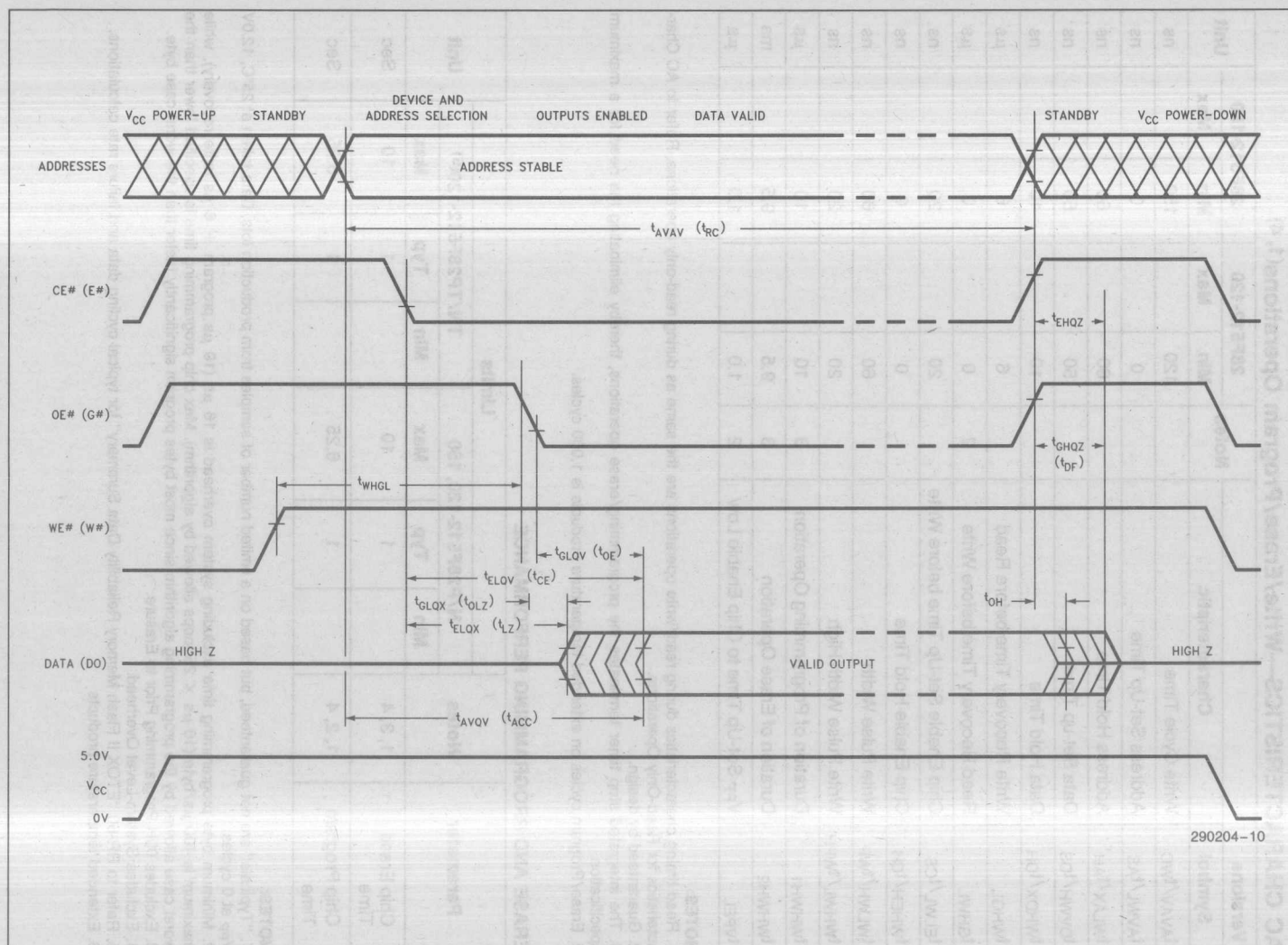
AC CHARACTERISTICS—Read-Only Operations

Versions(1)		Notes	N28F512-120 TN28F512-120 P28F512-120 TP28F512-120		N28F512-150 P28F512-150		Unit
Symbol	Characteristic		Min	Max	Min	Max	
t_{AVAV}/t_{RC}	Read Cycle Time		120		150		ns
t_{ELQV}/t_{CE}	Chip Enable Access Time			120		150	ns
t_{AVQV}/t_{ACC}	Address Access Time			120		150	ns
t_{GLQV}/t_{OE}	Output Enable Access Time			50		55	ns
t_{ELQX}/t_{LZ}	Chip Enable to Output in Low Z	2, 3	0		0		ns
t_{EHQZ}	Chip Disable to Output in High Z	2		55		55	ns
t_{GLQX}/t_{OLZ}	Output Enable to Output in Low Z	2, 3	0		0		ns
t_{GHQZ}/t_{DF}	Output Disable to Output in High Z	2		30		35	ns
t_{OH}	Output Hold from Address, CE #, or OE # Change	2, 4	0		0		ns
t_{WHGL}	Write Recovery Time before Read		6		6		μ s

NOTES:

1. Model number prefixes: N = PLCC, P = PDIP, T = Extended Temperature.
2. Sampled, not 100% tested.
3. Guaranteed by design.
4. Whichever occurs first.

Figure 6. AC Waveforms for Read Operations



290204-10

AC CHARACTERISTICS—Write/Erase/Program Operations^(1, 4)

Versions		Notes	28F512-120		28F512-150		Unit
Symbol	Characteristic		Min	Max	Min	Max	
t _{AVAV} /t _{WC}	Write Cycle Time		120		150		ns
t _{AVWL} /t _{AS}	Address Set-Up Time		0		0		ns
t _{WLAX} /t _{AH}	Address Hold Time		60		60		ns
t _{DVWH} /t _{DS}	Data Set-up Time		50		50		ns
t _{WHDX} /t _{DH}	Data Hold Time		10		10		ns
t _{WHGL}	Write Recovery Time before Read		6		6		μs
t _{GHWL}	Read Recovery Time before Write	2	0		0		μs
t _{ELWL} /t _{CS}	Chip Enable Set-Up Time before Write		20		20		ns
t _{WHEH} /t _{CH}	Chip Enable Hold Time		0		0		ns
t _{WLWH} /t _{WP}	Write Pulse Width		60		60		ns
t _{WHWL} /t _{WPH}	Write Pulse Width High		20		20		ns
t _{WHWH1}	Duration of Programming Operation	3	10		10		μs
t _{WHWH2}	Duration of Erase Operation	3	9.5		9.5		ms
t _{VPEL}	V _{PP} Set-Up Time to Chip Enable Low	2	1.0		1.0		μs

NOTES:

1. Read timing characteristics during read/write operations are the same as during read-only operations. Refer to AC Characteristics for Read-Only Operations.
2. Guaranteed by design.
3. The integrated stop timer terminates the programming/erase operations, thereby eliminating the need for a maximum specification.
4. Erase/Program cycles on extended temperature products is 1,000 cycles.

ERASE AND PROGRAMMING PERFORMANCE

Parameter	Notes	Limits						Unit
		N/P28F512-120, 150			TN/TP28F512-120 ⁽⁶⁾			
		Min	Typ	Max	Min	Typ	Max	
Chip Erase Time	1, 3, 4		1	10		1	10	Sec
Chip Program Time	1, 2, 4		1	6.25		1	6.25	Sec

NOTES:

1. "Typicals" are not guaranteed, but based on a limited number of samples from production lots. Data taken at 25°C, 12.0V V_{PP} at 0 cycles.
2. Minimum byte programming time excluding system overhead is 16 μs (10 μs program + 6 μs write recovery), while maximum is 400 μs/byte (16 μs × 25 loops allowed by algorithm). Max chip programming time is specified lower than the worst case allowed by the programming algorithm since most bytes program significantly faster than the worst case byte.
3. Excludes 00H Programming Prior to Erasure.
4. Excludes System-Level Overhead.
5. Refer to RR-60 "ETOX II Flash Memory Reliability Data Summary" for typical cycling data and failure rate calculations.
6. Extended temperature products

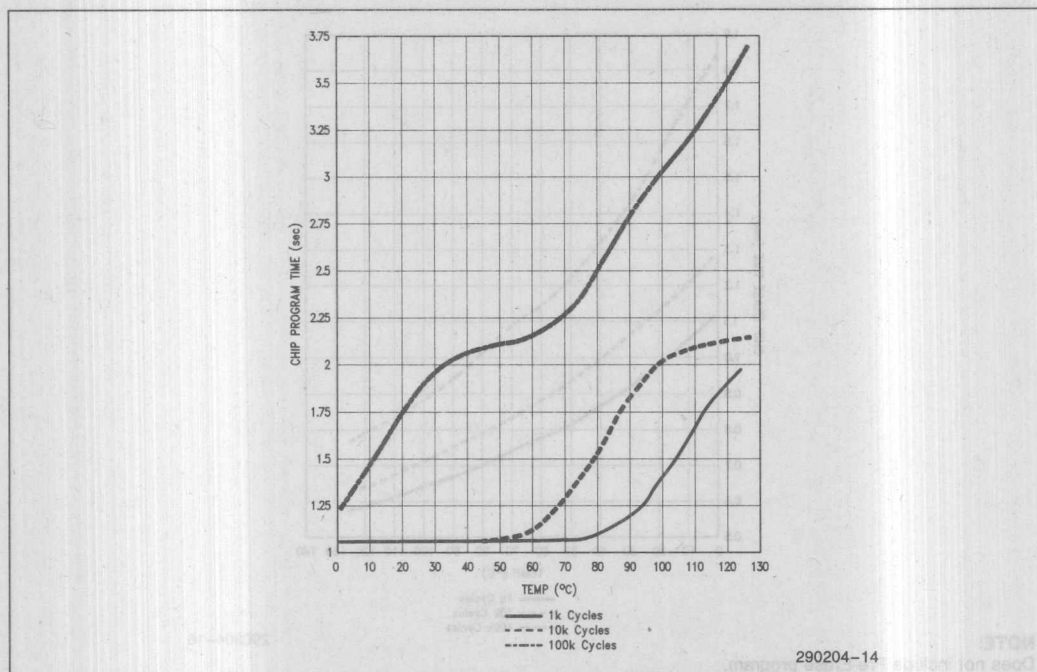


Figure 7. 28F512 Typical Program Time at 12V

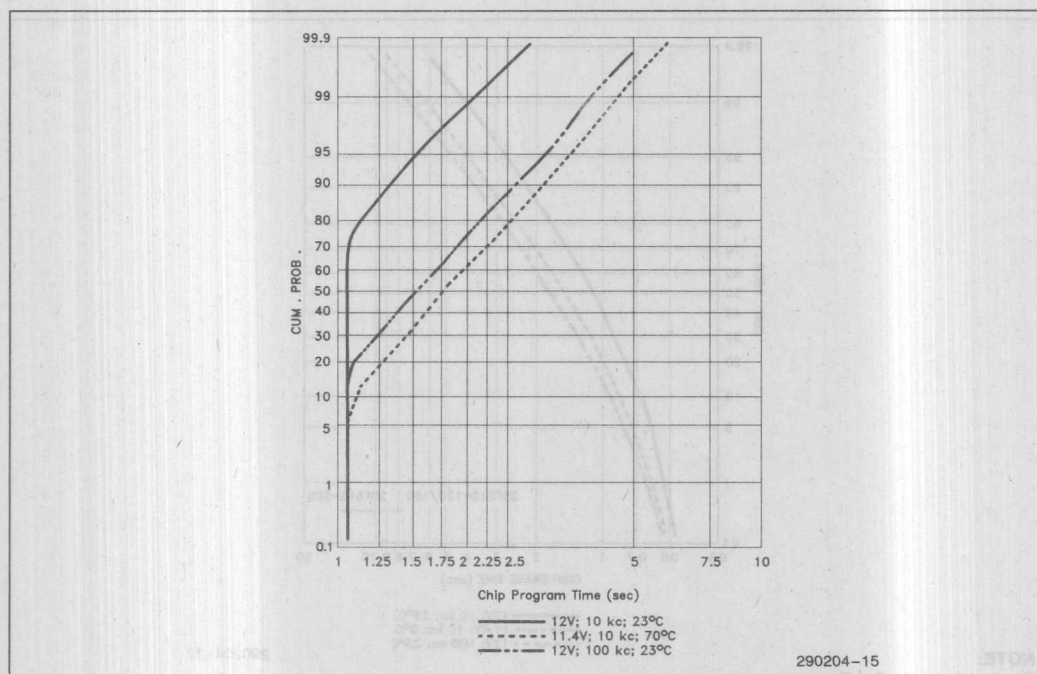
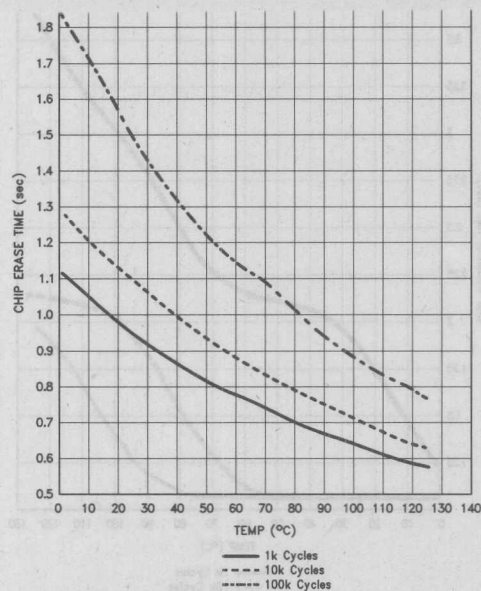


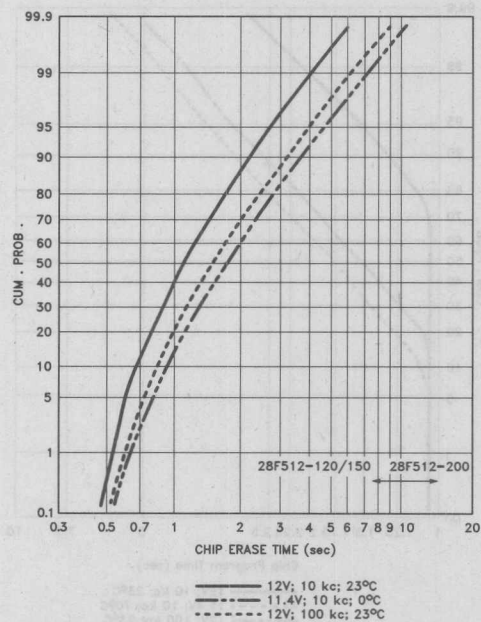
Figure 8. 28F512 Typical Programming Capability



NOTE:
Does not include Pre-Erase program.

290204-16

Figure 9. 28F512 Typical Erase Time at 12V



NOTE:
Does not include Pre-Erase program.

290204-17

Figure 10. 28F512 Typical Erase Capability

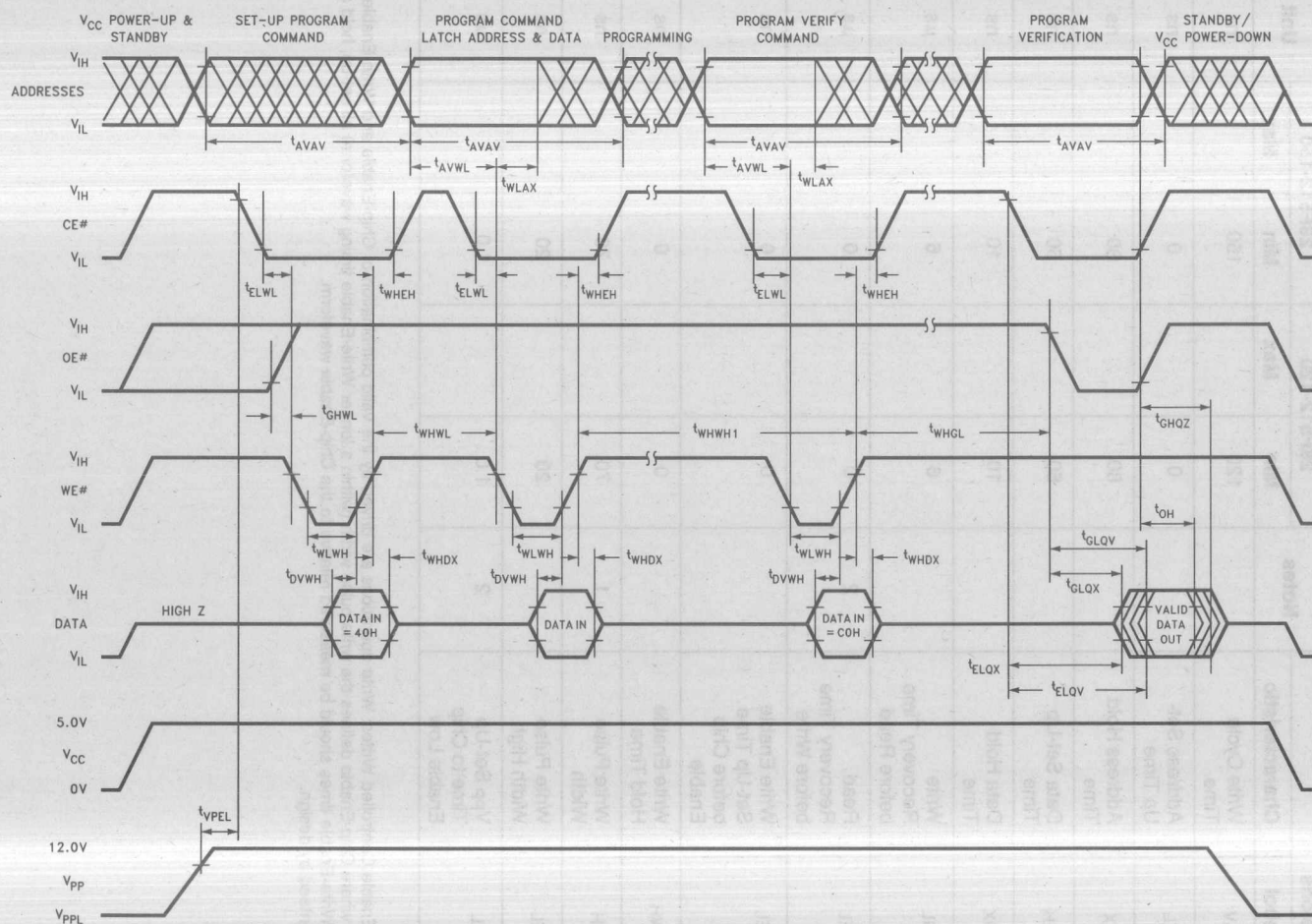


Figure 11. AC Waveforms for Programming Operations

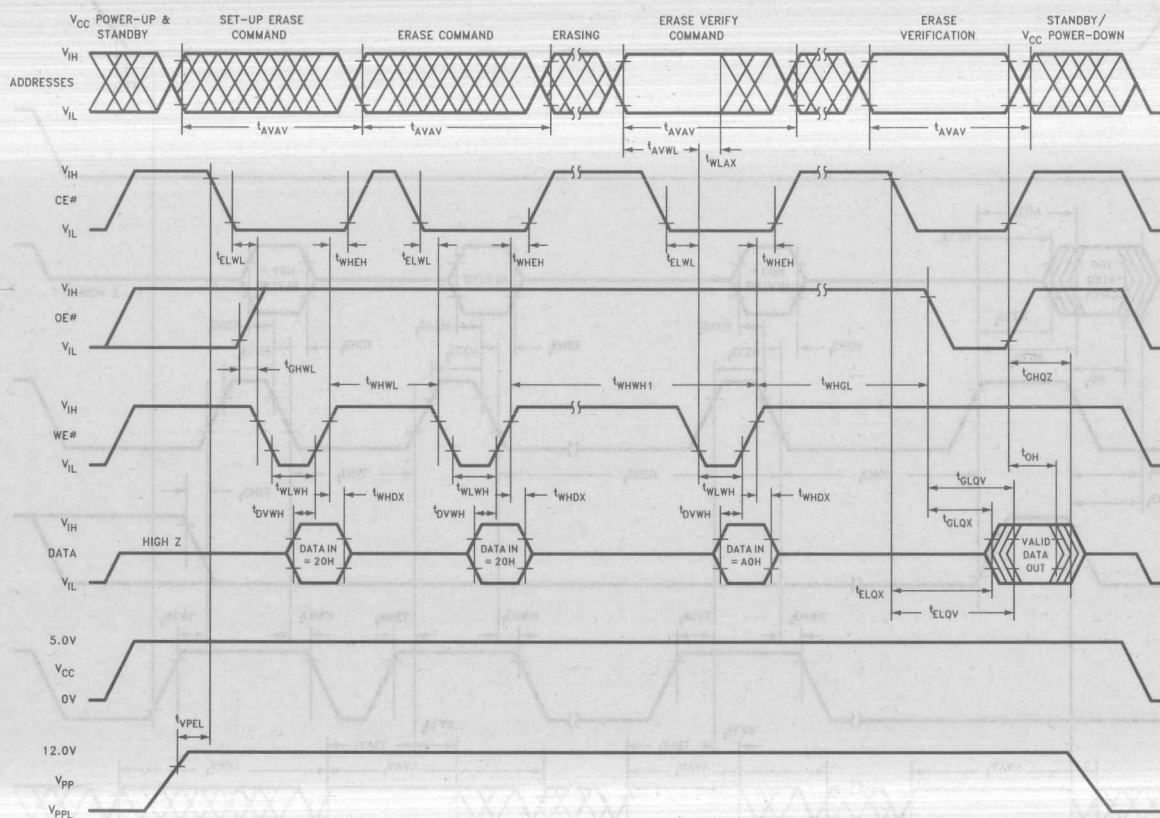
ALTERNATIVE CE#-CONTROLLED WRITES

Versions		Notes	28F512-120		28F512-150		Unit
Symbol	Characteristic		Min	Max	Min	Max	
t _{AVAV}	Write Cycle Time		120		150		ns
t _{AVEL}	Address Set-Up Time		0		0		ns
t _{ELAX}	Address Hold Time		80		80		ns
t _{DVEH}	Data Set-Up Time		50		50		ns
t _{EHDX}	Data Hold Time		10		10		ns
t _{EHGL}	Write Recovery Time before Read		6		6		μs
t _{GHEL}	Read Recovery Time before Write	2	0		0		μs
t _{WLEL}	Write Enable Set-Up Time before Chip Enable		0		0		ns
t _{EHWH}	Write Enable Hold Time		0		0		ns
t _{ELEH}	Write Pulse Width	1	70		70		ns
t _{EHXL}	Write Pulse Width High		20		20		ns
t _{VPEL}	V _{pp} Set-Up Time to Chip Enable Low	2	1.0		1.0		μs

NOTE:

1. Chip-Enable Controlled Writes: Write operations are driven by the valid combination of Chip-Enable and Write-Enable. In systems where Chip-Enable defines the write pulse width (within a longer Write-Enable timing waveform) all set-up, hold and inactive Write-Enable times should be measured relative to the Chip-Enable waveform.
2. Guaranteed by design.

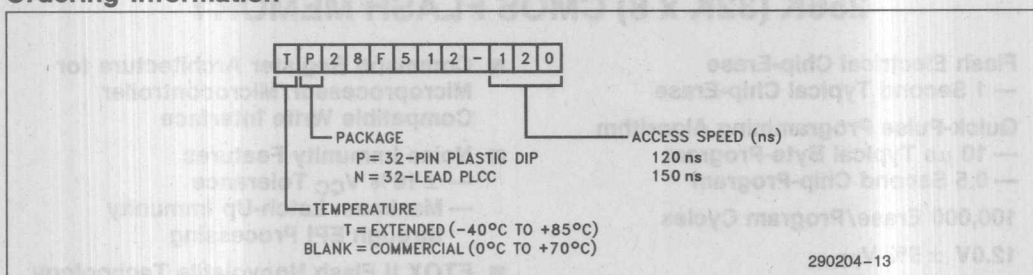
Figure 12. AC Waveforms for Erase Operations



290204-20



Ordering Information



Valid Combinations:

P28F512-120	N28F512-120	TP28F512-120
P28F512-150	N28F512-150	TN28F512-120

ADDITIONAL INFORMATION

ER-20, "ETOX II Flash Memory Technology"	294005
ER-24, "Intel Flash Memory"	294008
RR-60, "ETOX II Flash Memory Reliability Data Summary"	293002
AP-316, "Using Flash Memory for In-System Reprogrammable Nonvolatile Storage"	292046
AP-325 "Guide to Flash Memory Reprogramming"	292059

REVISION HISTORY

Number	Description
006	Removed 200 ns speed bin Revised Erase Maximum Pulse Count for Figure 5 from 3000 to 1000 Clarified AC and DC test conditions
007	Corrected AC Waveforms Added Extended Temperature devices; TP28F512-120, TN28F512-120
008	Revised symbols; i.e., \overline{CE} , \overline{OE} , etc. to CE#, OE#, etc.



28F256A 256K (32K x 8) CMOS FLASH MEMORY

- Flash Electrical Chip-Erase
 - 1 Second Typical Chip-Erase
- Quick-Pulse Programming Algorithm
 - 10 μ s Typical Byte-Program
 - 0.5 Second Chip-Program
- 100,000 Erase/Program Cycles
- 12.0V \pm 5% V_{pp}
- High-Performance Read
 - 120 ns Maximum Access Time
- CMOS Low Power Consumption
 - 10 mA Typical Active Current
 - 50 μ A Typical Standby Current
 - 0W Data Retention Power
- Integrated Program/Erase Stop Timer
- Command Register Architecture for Microprocessor/Microcontroller Compatible Write Interface
- Noise Immunity Features
 - \pm 10% V_{CC} Tolerance
 - Maximum Latch-Up Immunity through EPI Processing
- ETOX II Flash Nonvolatile Technology
 - EPROM-Compatible Process Base
 - High-Volume Manufacturing Experience
- JEDEC-Standard Pinouts
 - 32-Pin PDIP
 - 32-Lead PLCC

(See Packaging Spec., Order #231369)

Intel's 28F256A CMOS flash memory offers the most cost-effective and reliable alternative for read/write random access nonvolatile memory. The 28F256A adds electrical chip-erase and reprogramming to familiar EPROM technology. Memory contents can be rewritten: in a test socket; in a PROM-programmer socket; on-board during subassembly test; in-system during final test; and in-system after-sale. The 28F256A increases memory flexibility, while contributing to time and cost savings.

The 28F256A is a 256-kilobit nonvolatile memory organized as 32,768 bytes of 8 bits. Intel's 28F256A is offered in 32-pin plastic dip and 32-lead PLCC. Pin assignments conform to JEDEC standards.

Extended erase and program cycling capability is designed into Intel's ETOX II (EPROM Tunnel Oxide) process technology. Advanced oxide processing, an optimized tunneling structure, and lower electric field combine to extend reliable cycling beyond that of traditional EEPROMs. With the 12.0V V_{pp} supply, the 28F256A performs a minimum of 10,000 erase and program cycles well within the time limits of the Quick-Pulse Programming and Quick-Erase algorithms.

Intel's 28F256A employs advanced CMOS circuitry for systems requiring high-performance access speeds, low power consumption, and immunity to noise. Its 120 ns access time provides no-WAIT-state performance for a wide range of microprocessors and microcontrollers. Typical standby current of 50 μ A translates into power savings when the device is deselected. Finally, the highest degree of latch-up protection is achieved through Intel's unique EPI processing. Prevention of latch-up is provided for stresses up to 100 mA on address and data pins, from $-1V$ to V_{CC} + 1V.

With Intel's ETOX II process base, the 28F256A levers years of EPROM experience to yield the highest levels of quality, reliability, and cost-effectiveness.

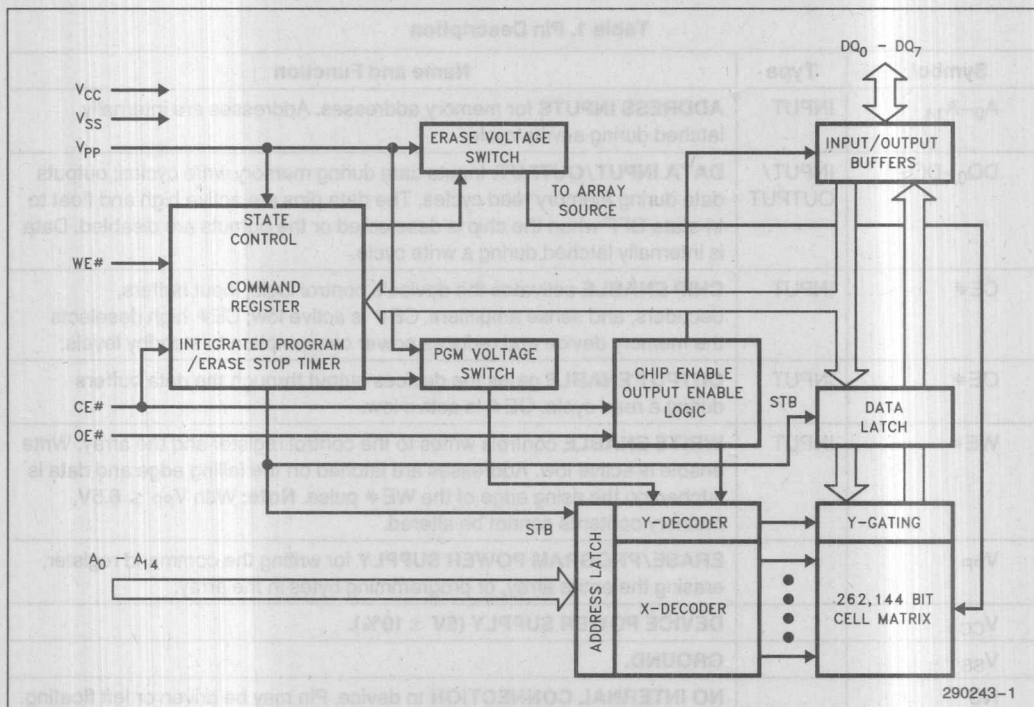


Figure 1. 28F256A Block Diagram

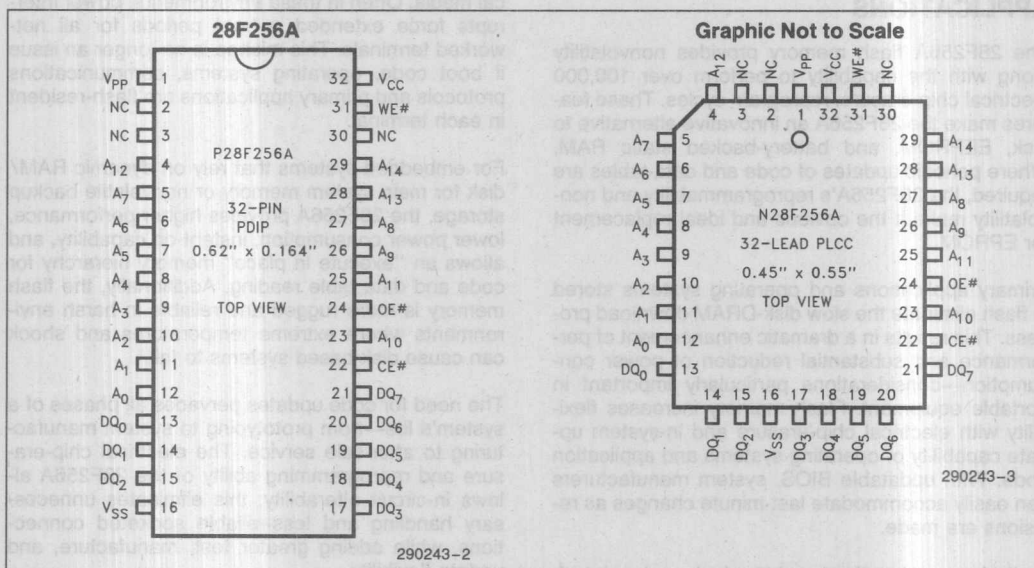


Figure 2. 28F256A Pin Configurations

Table 1. Pin Description

Symbol	Type	Name and Function
A ₀ –A ₁₄	INPUT	ADDRESS INPUTS for memory addresses. Addresses are internally latched during a write cycle.
DQ ₀ –DQ ₇	INPUT/ OUTPUT	DATA INPUT/OUTPUT: Inputs data during memory write cycles; outputs data during memory read cycles. The data pins are active high and float to tri-state OFF when the chip is deselected or the outputs are disabled. Data is internally latched during a write cycle.
CE #	INPUT	CHIP ENABLE activates the device's control logic, input buffers, decoders, and sense amplifiers. CE # is active low; CE # high deselects the memory device and reduces power consumption to standby levels.
OE #	INPUT	OUTPUT ENABLE gates the devices output through the data buffers during a read cycle. OE # is active low.
WE #	INPUT	WRITE ENABLE controls writes to the control register and the array. Write enable is active low. Addresses are latched on the falling edge and data is latched on the rising edge of the WE # pulse. Note: With V _{pp} ≤ 6.5V, memory contents cannot be altered.
V _{pp}		ERASE/PROGRAM POWER SUPPLY for writing the command register, erasing the entire array, or programming bytes in the array.
V _{CC}		DEVICE POWER SUPPLY (5V ± 10%).
V _{SS}		GROUND.
NC		NO INTERNAL CONNECTION to device. Pin may be driven or left floating.

APPLICATIONS

The 28F256A flash memory provides nonvolatility along with the capability to perform over 100,000 electrical chip-erase/reprogram cycles. These features make the 28F256A an innovative alternative to disk, EEPROM, and battery-backed static RAM. Where periodic updates of code and data-tables are required, the 28F256A's reprogrammability and nonvolatility make it the obvious and ideal replacement for EPROM.

Primary applications and operating systems stored in flash eliminate the slow disk-DRAM download process. This results in a dramatic enhancement of performance and substantial reduction of power consumption—considerations particularly important in portable equipment. Flash memory increases flexibility with electrical chip-erase and in-system update capability of operating systems and application code. With updatable BIOS, system manufacturers can easily accommodate last-minute changes as revisions are made.

In diskless workstations and terminals, network traffic reduces to a minimum and systems become instant-on. Reliability exceeds that of electromechanical

media. Often in these environments, power interrupts force extended re-boot periods for all networked terminals. This mishap is no longer an issue if boot code, operating systems, communications protocols and primary applications are flash-resident in each terminal.

For embedded systems that rely on dynamic RAM/disk for main system memory or nonvolatile backup storage, the 28F256A provides higher performance, lower power consumption, instant-on capability, and allows an "execute in place" memory hierarchy for code and data table reading. Additionally, the flash memory is more rugged and reliable in harsh environments where extreme temperatures and shock can cause disk-based systems to fail.

The need for code updates pervades all phases of a system's life—from prototyping to system manufacturing to after-sale service. The electrical chip-erase and reprogramming ability of the 28F256A allows in-circuit alterability; this eliminates unnecessary handling and less-reliable socketed connections, while adding greater test, manufacture, and update flexibility.

Material and labor costs associated with code changes increases at higher levels of system inte-

gration—the most costly being code updates after sale. Code “bugs”, or the desire to augment system functionality, prompt after-sale code updates. Field revision to EPROM-based code requires the removal of EPROM components or entire boards. With the 28F256A, code updates are implemented locally via an edge-connector, or remotely over a communications link.

For systems currently using a high-density static RAM/battery configuration for data accumulation, flash memory’s inherent nonvolatility eliminates the need for battery backup. The concern for battery failure no longer exists, an important consideration for portable equipment and medical instruments, both requiring continuous performance. In addition, flash memory offers a considerable cost advantage over static RAM.

Flash memory’s electrical chip-erase, byte programmability and complete nonvolatility fit well with data accumulation and recording needs. Electrical

chip-erase gives the designer a “blank slate” in which to log or record data. Data can be periodically off-loaded for analysis and the flash memory erased producing a new “blank slate”.

A high degree of on-chip feature integration simplifies memory-to-processor interfacing. Figure 3 depicts two 28F256As tied to the 80C186 system bus. The 28F256A’s architecture minimizes interface circuitry needed for complete in-circuit updates of memory contents.

With cost-effective in-system reprogramming, extended cycling capability, and true nonvolatility, the 28F256A is a functional superset of one or more of the alternatives: EPROMs, EEPROMs, battery backed static RAM, or disk. EPROM-compatible read specifications, straightforward interfacing, and in-circuit alterability offer designers unlimited flexibility to meet the high standards of today’s designs.

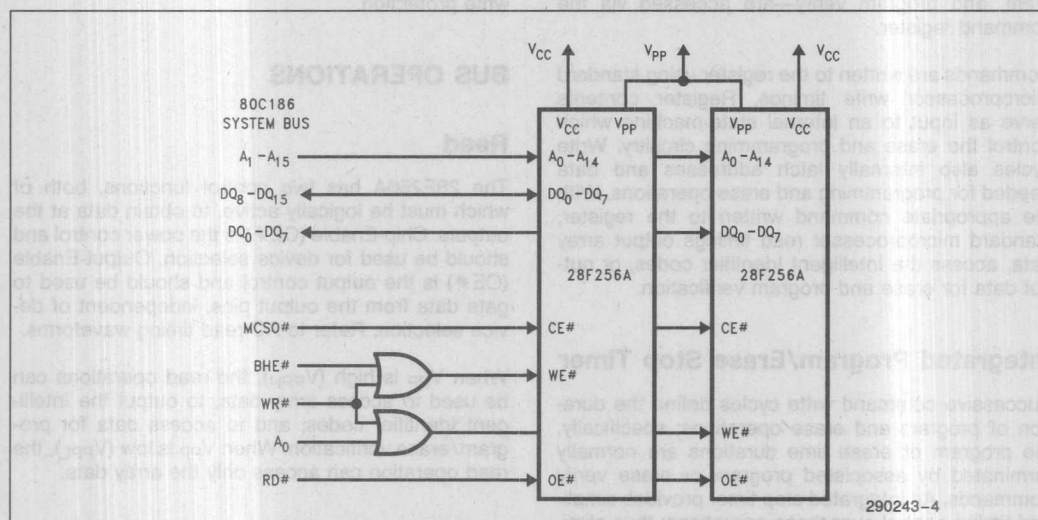


Figure 3. 28F256As in a 80C186 System

PRINCIPLES OF OPERATION

Flash memory augments EPROM functionality with in-circuit electrical erasure and reprogramming. The 28F256A introduces a command register to manage this new functionality. The command register allows for: 100% TTL-level control inputs; fixed power supply during erasure and programming; and maximum EPROM compatibility.

In the absence of high voltage on the V_{PP} pin, the 28F256A is a read-only memory. Manipulation of the external memory-control pins yields the standard EPROM read, standby, output disable, and Intelligent Identifier operations.

The same EPROM read, standby, and output disable operations are available when high voltage is applied to the V_{PP} pin. In addition, high voltage on V_{PP} enables erasure and programming of the device. All functions associated with altering memory contents—Intelligent Identifier, erase, erase verify, program, and program verify—are accessed via the command register.

Commands are written to the register using standard microprocessor write timings. Register contents serve as input to an internal state-machine which control the erase and programming circuitry. Write cycles also internally latch addresses and data needed for programming and erase operations. With the appropriate command written to the register, standard microprocessor read timings output array data, access the Intelligent Identifier codes, or output data for erase and program verification.

Integrated Program/Erase Stop Timer

Successive command write cycles define the duration of program and erase operations; specifically, the program or erase time durations are normally terminated by associated program or erase verify commands. An integrated stop timer provides simplified timing control over these operations; thus eliminating the need for maximum program/erase timing specifications. Programming and erase pulse durations are minimums only. When the stop timer terminates a program or erase operation, the device enters an inactive state and remains inactive until receiving the appropriate verify or reset command.

Write Protection

The command register is only active when V_{PP} is at high voltage. Depending upon the application, the system designer may choose to make the V_{PP} power supply switchable—available only when memory updates are desired. When $V_{PP} = V_{PPL}$, the contents of the register default to the read command, making the 28F256A a read-only memory. In this mode, the memory contents cannot be altered.

Or, the system designer may choose to "hardwire" V_{PP} , making the high voltage supply constantly available. In this case, all Command Register functions are inhibited whenever V_{CC} is below the write lockout voltage V_{LKO} . (See Power Up/Down Protection). The 28F256A is designed to accommodate either design practice, and to encourage optimization of the processor-memory interface.

The two-step program/erase write sequence to the Command Register provides additional software write protection.

BUS OPERATIONS

Read

The 28F256A has two control functions, both of which must be logically active, to obtain data at the outputs. Chip-Enable ($CE\#$) is the power control and should be used for device selection. Output-Enable ($OE\#$) is the output control and should be used to gate data from the output pins, independent of device selection. Refer to AC read timing waveforms.

When V_{PP} is high (V_{PPH}), the read operations can be used to access array data, to output the Intelligent Identifier codes, and to access data for program/erase verification. When V_{PP} is low (V_{PPL}), the read operation can access only the array data.

Output Disable

With Output-Enable at a logic-high level (V_{IH}), output from the device is disabled. Output pins are placed in a high-impedance state.

Standby

With Chip-Enable at a logic-high level, the standby operation disables most of the 28F256A's circuitry and substantially reduces device power consumption. The outputs are placed in a high-impedance

state, independent of the Output-Enable signal. If the 28F256A is deselected during erasure, programming, or program/erase verification, the device draws active current until the operation is terminated.

Intelligent Identifier Operation

The Intelligent Identifier operation outputs the manufacturer code (89H) and device code (B9H). Programming equipment automatically matches the device with its proper erase and programming algorithms. With Chip-Enable and Output-Enable at a logic low level, rising A_9 to high voltage V_{ID} (see D.C. Characteristics) activates the operation. Data read from locations 0000H and 0001H represent the manufacturer's code and the device code, respectively.

The manufacturer- and device-codes can also be read via the command register, for instances where the 28F256A is erased and reprogrammed in the target system. Following a write of 90H to the command register, a read from address location 0000H outputs the manufacturer code (89H). A read from address 0001H outputs the device code (B9H).

Write

Device erasure and programming are accomplished via the command register, when high voltage is ap-

plied to the V_{pp} pin. The contents of the register serve as input to the internal state-machine. The state-machine outputs dictate the function of the device.

The command register itself does not occupy an addressable memory location. The register is a latch used to store the command, along with address and data information needed to execute the command.

The command register is written by bringing Write-Enable to a logic-low level (V_{IL}), while Chip-Enable is low. Addresses are latched on the falling edge of Write-Enable, while data is latched on the rising edge of the Write-Enable pulse. Standard microprocessor write timings are used.

Refer to AC Write Characteristics and the Erase/Programming Waveforms for specific timing parameters.

COMMAND DEFINITIONS

When low voltage is applied to the V_{pp} pin, the contents of the command register default to 00H, enabling read-only operations.

Placing high voltage on the V_{pp} pin enables read/write operations. Device operations are selected by writing specific data patterns into the command register. Table 3 defines these 28F256A register commands.

5

Table 2. 28F256A Bus Operations

		Pins	$V_{pp}(1)$	A_0	A_9	CE #	OE #	WE #	DQ ₀ -DQ ₇
Operation									
READ-ONLY	Read		V_{PPL}	A_0	A_9	V_{IL}	V_{IL}	V_{IH}	Data Out
	Output Disable		V_{PPL}	X(7)	X	V_{IL}	V_{IH}	V_{IH}	Tri-State
	Standby		V_{PPL}	X	X	V_{IH}	X	X	Tri-State
	Intelligent ID Manufacturer(2)		V_{PPL}	V_{IL}	$V_{ID}(3)$	V_{IL}	V_{IL}	V_{IH}	Data = 89H
	Intelligent ID Device(2)		V_{PPL}	V_{IH}	$V_{ID}(3)$	V_{IL}	V_{IL}	V_{IH}	Data = B9H
READ/ WRITE	Read		V_{PPH}	A_0	A_9	V_{IL}	V_{IL}	V_{IH}	Data Out(4)
	Output Disable		V_{PPH}	X	X	V_{IL}	V_{IH}	V_{IH}	Tri-State
	Standby(5)		V_{PPH}	X	X	V_{IH}	X	X	Tri-State
	Write		V_{PPH}	A_0	A_9	V_{IL}	V_{IH}	V_{IL}	Data In (6)

NOTES:

1. Refer to DC Characteristics. When $V_{pp} = V_{PPL}$ memory contents can be read but not written or erased.
2. Manufacturer and device codes may also be accessed via a command register write-sequence. Refer to Table 3. All other addresses low.
3. V_{ID} is the Intelligent Identifier high voltage. Refer to D.C. Characteristics.
4. Read operations with $V_{pp} = V_{PPH}$ may access array data or the Intelligent Identifier codes.
5. With V_{pp} at high voltage, the standby current equals $I_{CC} + I_{pp}$ (standby).
6. Refer to Table 3 for valid Data-In during a write operation.
7. X can be V_{IL} or V_{IH} .

Table 3. Command Definitions

Command	Bus Cycles Req'd	First Bus Cycle			Second Bus Cycle		
		Operation ⁽¹⁾	Address ⁽²⁾	Data ⁽³⁾	Operation ⁽¹⁾	Address ⁽²⁾	Data ⁽³⁾
Read Memory	1	Write	X	00H			
Read Intelligent ID Codes	3	Write	X	90H	Read	(4)	(4)
Set-Up Erase/Erase ⁽⁶⁾	2	Write	X	20H	Write	X	20H
Erase Verify ⁽⁶⁾	2	Write	EA	A0H	Read	X	EVD
Set-Up Program/Program ⁽⁵⁾	2	Write	X	40H	Write	PA	PD
Program Verify ⁽⁵⁾	2	Write	X	C0H	Read	X	PVD
Reset ⁽⁷⁾	2	Write	X	FFH	Write	X	FFH

NOTES:

- Bus operation are defined in Table 2.
- IA = Identifier address: 00H for manufacturer code, 01H for device code.
EA = Address of memory location to be read during erase verify.
PA = Address of memory location to be programmed.
Addresses are latched on the falling edge of the Write-Enable pulse.
- ID = Data read from location IA during device identification. (Mfr = 89H, Device = B9H).
EVD = Data read from location EA during erase verify.
PD = Data to be programmed at location PA. Data is latched on the rising edge of the Write-Enable.
PVD = Data read from location PA during program verify. PA is latched on the Program command.
- Following the Read Intelligent ID command, two read operations access manufacturer and device codes.
- Figure 4 illustrates the Quick-Pulse Programming Algorithm.
- Figure 5 illustrates the Quick-Erase Algorithm.
- The second bus cycle must be followed by the desired command register write.

Read Command

While V_{PP} is high, for erasure and programming, memory contents can be accessed via the read command. The read operation is initiated by writing 00H into the command register. Microprocessor read cycles retrieve array data. The device remains enabled for reads until the command register contents are altered.

The default contents of the register upon V_{PP} power-up is 00H. This default value ensures that no spurious alternation of memory contents occurs during the V_{PP} power transition. Where the V_{PP} supply is hard-wired to the 28F256A, the device powers-up and remains enabled for reads until the command register contents are changed. Refer to the AC Read Characteristics and Waveforms for specific timing parameters.

Intelligent Identifier Command

Flash memories are intended for use in applications where the local CPU alters memory contents. As such, manufacturer- and device-codes must be accessible while the device resides in the target system. PROM programmers typically access signature codes by raising A_9 to a high voltage. However, mul-

tiplexing high voltage onto address lines is not a desired system-design practice.

The 28F256A contains an Intelligent Identifier operation to supplement traditional PROM-programming methodology. The operation is initiated by writing 90H into the command register. Following the command write, a read cycle from address 0000H retrieves the manufacturer code 89H. A read cycle from address 0001H returns the device code B9H. To terminate the operation, it is necessary to write another valid command into the register.

Set-Up Erase/Erase Commands

Set-up Erase is a command-only operation that stages the device for electrical erase of all bytes in the array. The set-up erase operation is performed by writing 20H to the command register. To commence chip-erasure, the erase command (20H) must again be written to the register. The erase operation begins with the rising edge of the Write-Enable pulse and terminate with the rising edge of the next Write-Enable pulse (i.e., Erase-Verify Command).

This two-step sequence of set-up followed by execution ensures that memory contents are not accidentally erased. Also, chip-erasure can only occur when high voltage is applied to the V_{PP} pin. In the absence

of this high voltage, memory contents are protected against erasure. Refer to AC Erase Characteristics and Waveforms for specific timing parameters.

Erase-Verify Command

The erase command erases all of the bytes of the array in parallel. After each erase operation, all bytes must be verified. The erase verify operation is initiated by writing A0H into the command register. The address for the byte to be verified must be supplied as it is latched on the falling edge of the Write-Enable pulse. The register write terminates the erase operation with the rising edge of its Write-Enable pulse.

The 28F256A applies an internally-generated margin voltage to the addressed byte. Reading FFH from the addressed byte indicates that all bits in the byte are erased.

The erase-verify command must be written to the command register prior to each byte verification to latch its address. The process continues for each byte in the array until a byte does not return FFH data, or the last address is accessed.

In the case where the data read is not FFH, another erase operation is performed. (Refer to Set-Up Erase/Erase.) Verification then resumes from the address of the last-verified byte. Once all bytes in the array have been verified, the erase step is complete. The device can be programmed. At this point, the verify operation is terminated by writing a valid command (e.g., Program Set-Up) to the command register. Figure 5, the Quick-Erase algorithm, illustrates how commands and bus operations are combined to perform electrical erasure of the 28F256A. Refer to AC Erase Characteristics and Waveforms for specific timing parameters.

Set-Up Program/Program Commands

Set-up program is a command-only operation that stages the device for byte programming. Writing 40H into the command register performs the set-up operation.

Once the program set-up operation is performed, the next Write-Enable pulse causes a transition to an active programming operation. Addresses are internally latched on the falling edge of the Write-Enable pulse. Data is internally latched on the rising edge of the Write-Enable pulse. The rising edge of Write-Enable also begins the programming operation. The programming operation terminates with the next rising edge of Write-Enable, used to write the

program-verify command. Refer to AC Programming Characteristics and Waveforms for specific timing parameters.

Program Verify Command

The 28F256A is programmed on a byte-by-byte basis. Byte programming may occur sequentially or at random. Following each programming operation, the byte just programmed must be verified.

The program-verify operation is initiated by writing C0H into the command register. The register write terminates the programming operation with the rising edge of its Write-Enable pulse. The program-verify operation stages the device for verification of the byte last programmed. No new address information is latched.

The 28F256A applies an internally-generated margin voltage to the byte. A microprocessor read cycle outputs the data. A successful comparison between the programmed byte and true data means that the byte is successfully programmed. Programming then proceeds to the next desired byte location. Figure 4, the 28F256A Quick-Pulse Programming algorithm, illustrates how commands are combined with bus operations to perform byte programming. Refer to AC Programming Characteristics and Waveforms for specific timing parameters.

Reset Command

A reset command is provided as a means to safely abort the erase- or program-command sequences. Following either set-up command (erase or program) with two consecutive writes of FFH will safely abort the operation. Memory contents will not be altered. A valid command must then be written to place the device in the desired state.

EXTENDED ERASE/PROGRAM CYCLING

EEPROM cycling failures have always concerned users. The high electrical field required by thin oxide EEPROMs for tunneling can literally tear apart the oxide at defect regions. To combat this, some suppliers have implemented redundancy schemes, reducing cycling failures to insignificant levels. However, redundancy requires that cell size be doubled—an expensive solution.

Intel has designed extended cycling capability into its ETOX II flash memory technology. Resulting improvements in cycling reliability come without increasing memory cell size or complexity. First, an

advanced tunnel oxide increases the charge carrying ability ten-fold. Second, the oxide area per cell subjected to the tunneling electric field is one-tenth that of common EEPROMs, minimizing the probability of oxide defects in the region. Finally, the peak electric field during erasure is approximately 2 MV/cm lower than EEPROM. The lower electric field greatly reduces oxide stress and the probability of failure—increasing time to wear out by a factor of 100,000,000.

The 28F256A is capable of 100,000 program/erase cycles. The device is programmed and erased using Intel's Quick-Pulse Programming and Quick-Erase algorithms. Intel's algorithmic approach uses a series of operations (pulses), along with byte verification, to completely and reliably erase and program the device.

For further reliability information, see Reliability Report RR-60 (ETOX II Reliability Data Summary).

QUICK-PULSE PROGRAMMING ALGORITHM

The Quick-Pulse Programming algorithm uses programming operations of 10 μ s duration. Each operation is followed by a byte verification to determine when the addressed byte has been successfully programmed. The algorithm allows for up to 25 programming operations per byte, although most bytes verify on the first or second operation. The entire sequence of programming and byte verification is

performed with V_{pp} at high voltage. Figure 4 illustrates the Quick-Pulse Programming algorithm.

QUICK-ERASE ALGORITHM

Intel's Quick-Erase algorithm yields fast and reliable electrical erasure of memory contents. The algorithm employs a closed-loop flow, similar to the Quick-Pulse Programming algorithm, to simultaneously remove charge from all bits in the array. Erasure begins with a read of memory contents. The 28F256A is erased when shipped from the factory. Reading FFH data from the device would immediately be followed by device programming.

For devices being erased and reprogrammed, uniform and reliable erasure is ensured by first programming all bits in the device to their charged state (data = 00H). This is accomplished, using the Quick-Pulse Programming algorithm, in approximately one-half second.

Erase execution then continues with an initial erase operation. Erase verification (data = FFH) begins at address 0000H and continues through the array to the last address, or until data other than FFH is encountered. With each erase operation, an increasing number of bytes verify to the erased state. Erase efficiency may be improved by storing the address of the last byte verified in a register. Following the next erase operation, verification starts at that stored address location. Erasure typically occurs in one second. Figure 5 illustrates the Quick-Erase algorithm.

A reset command is provided as a means to safely abort the erase or program command sequence. Following either set-up command (erase or program) with two consecutive writes of FFH will safely abort the operation. Memory contents will not be altered. A valid command must first be written to place the device in the desired state.

EXTENDED ERASE/PROGRAM CYCLING

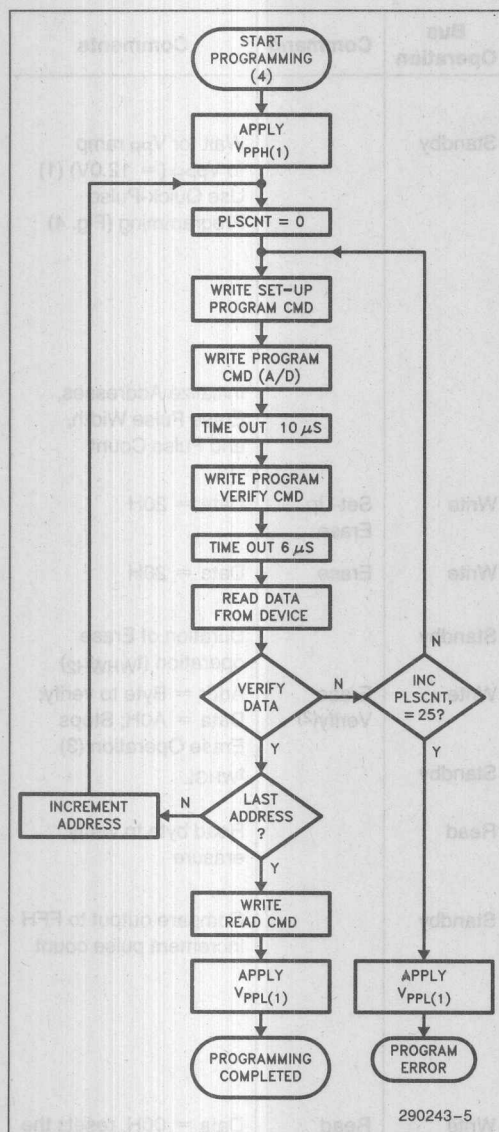
EEPROM cycling failures have always been a concern. The high electrical field required by the oxide used in EEPROMs for tunneling can create local stress points at defect regions. To combat this, some suppliers have implemented redundancy schemes, requiring cycling failures to be ignored. However, redundancy requires that data be copied into an expensive solution.

Intel has designed extended cycling capability into its ETOX II flash memory technology. Resulting improvements in cycling reliability come without increasing memory cell size or complexity. First, an

Set-Up Program/Program Commands

Set-up program is a command-only operation that erases the device for byte programming. When 40H is written to the command register, the set-up operation begins.

Once the program set-up operation is performed, the next Write-Enable pulse causes a transition to an active programming operation. Addresses are internally latched on the falling edge of the Write-Enable pulse. Data is internally latched on the rising edge of the Write-Enable pulse. The rising edge of the Write-Enable pulse begins the programming operation. The programming operation terminates with the next falling edge of Write-Enable, used to write the

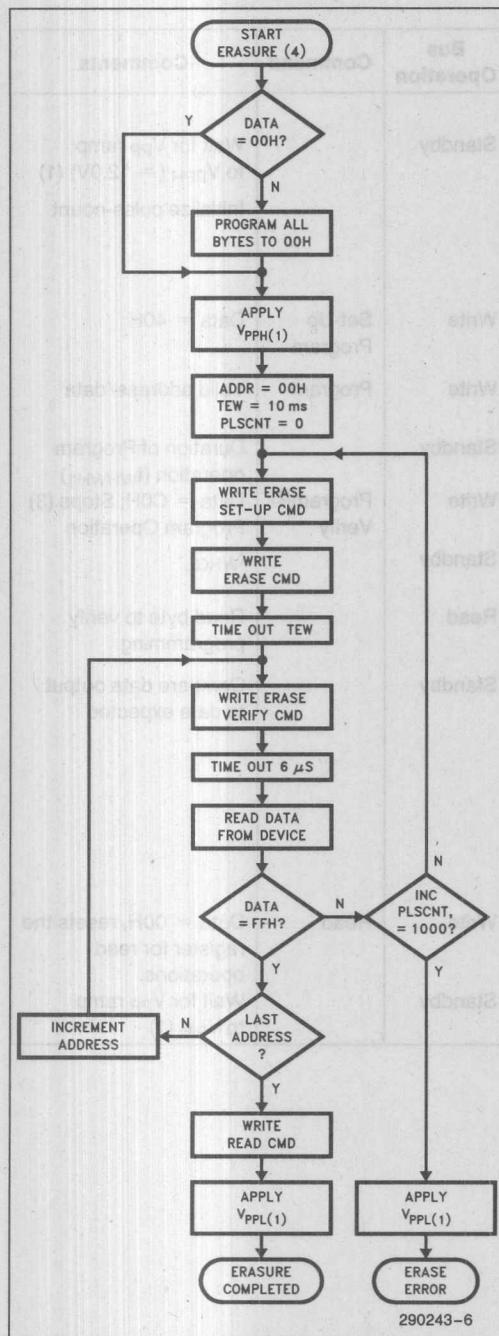


NOTES:

1. See DC Characteristics for the value of V_{ppH} and V_{pPL} .
2. Program Verify is only performed after byte programming. A final read/compare may be performed (optional) after the register is written with the Read command.
3. Refer to principles of operation.
4. CAUTION: The algorithm MUST BE FOLLOWED to ensure proper and reliable operation of the device.

Figure 4. 28F256A Quick-Pulse Programming Algorithm

Bus Operation	Command	Comments
Standby		Wait for V_{pp} ramp to V_{ppH} (= 12.0V) (1) Initialize pulse-count
Write	Set-Up Program	Data = 40H
Write	Program	Valid address/data
Standby		Duration of Program operation (t_{WHWH1})
Write	Program(2) Verify	Data = C0H; Stops (3) Program Operation
Standby		t_{WHGL}
Read		Read byte to verify programming
Standby		Compare data output to data expected
Write	Read	Data = 00H, resets the register for read operations.
Standby		Wait for V_{pp} ramp to V_{pPL} (1)

**NOTES:**

1. See DC Characteristics for the value of V_{PPH} and V_{PPL} .
2. Erase Verify is performed only after chip-erase. A final read/compare may be performed (optional) after the register is written with the Read command.

3. Refer to principles of operation.

4. CAUTION: The algorithm MUST BE FOLLOWED to ensure proper and reliable operation of the device.

Bus Operation	Command	Comments
Standby		Wait for V_{PP} ramp to V_{PPH} (= 12.0V) (1) Use Quick-Pulse Programming (Fig. 4)
Write	Set-Up Erase	Data = 20H Initialize Addresses, Erase Pulse Width, and Pulse Count
Write	Erase	Data = 20H
Standby		Duration of Erase operation (t_{WHWH2})
Write	Erase Verify(2)	Addr = Byte to verify; Data = A0H; Stops Erase Operation (3)
Standby		t_{WHGL}
Read		Read byte to verify erasure
Standby		Compare output to FFH increment pulse count
Write	Read	Data = 00H, resets the register for read operations.
Standby		Wait for V_{PP} ramp to V_{PPL} (1)

Figure 5. 28F256A Quick-Erase Algorithm

DESIGN CONSIDERATIONS

Two-Line Output Control

Flash memories are often used in larger memory arrays. Intel provides two read-control inputs to accommodate multiple memory connections. Two-line control provides for:

- the lowest possible memory power dissipation, and
- complete assurance that output bus contention will not occur.

To efficiently use these two control units, an address-decoder output should drive chip-enable, while the system's read signal controls all flash memories and other parallel memories. This assures that only enabled memory devices have active outputs, while deselected devices maintain the low power standby condition.

Power Supply Decoupling

Flash memory power-switching characteristics require careful device decoupling. System designers are interested in three supply current (I_{CC}) issues—standby, active, and transient current peaks produced by falling and rising edges of chip-enable. The capacitive and inductive loads on the device outputs determine the magnitudes of these peaks.

Two-line control and proper decoupling capacitor selection will suppress transient voltage peaks. Each device should have a 0.1 μ F ceramic capacitor connected between V_{CC} and V_{SS} , and between V_{PP} and V_{SS} .

Place the high-frequency, low-inherent-inductance capacitors as close as possible to the devices. Also, for every eight devices, a 4.7 μ F electrolytic capacitor should be placed at the array's power supply connection, between V_{CC} and V_{SS} . The bulk capacitor will overcome voltage slumps caused by printed-circuit-board trace inductance, and will supply charge to the smaller capacitors as needed.

V_{PP} Trace on Printed Circuit Boards

Programming flash memories, while they reside in the target smith, requires that the printed circuit board designer pay attention to the V_{PP} pin power supply trace. Use similar trace widths and layout considerations given the V_{CC} power bus. Adequate V_{PP} supply traces and decoupling will decrease V_{PP} voltage spikes and overshoots.

Power Up/Down Protection

The 28F256A is designed to offer protection against accidental erasure or programming during power transitions. Upon power-up, the 28F256A is indifferent as to which power supply, V_{PP} or V_{CC} , powers up first. **Power supply sequencing is not required.** Internal circuitry in the 28F256A ensures that the command register is reset to the read mode upon power up.

A system designer must guard against active writes for V_{CC} voltages above V_{LKO} when V_{PP} is active. Since both $WE\#$ and $CE\#$ must be low for a command write, driving either to V_{IH} will inhibit writes. The control register architecture provides an added level of protection since alteration of memory contents only occurs after successful completion of the two-step command sequences.

28F256A Power Dissipation

When designing portable systems, designers must consider battery power consumption not only during device operation, but also for data retention during system idle time. Flash nonvolatility increases the usable battery life of your system because the 28F256A does not consume any power to retain code or data when the system is off. Table 4 illustrates the power dissipated when updating the 28F256A.

Table 4. 28F256A Typical Update Power Dissipation⁽⁴⁾

Operation	Power Dissipation (Watt-Seconds)	Notes
Array Program/Program Verify	0.043	1
Array Erase/Erase Verify	0.083	2
One Complete Cycle	0.169	3

NOTES:

- Formula to calculate typical Program/Program Verify Power = $[V_{PP} \times \# \text{ Bytes} \times \text{typical } \# \text{ Prog Pulses} (t_{WHWH1} \times I_{PP2} \text{ typical} + t_{WHGL} \times I_{PP4} \text{ typical})] + [V_{CC} \times \# \text{ Bytes} \times \text{typical } \# \text{ Prog Pulses} (t_{WHWH1} \times I_{CC2} \text{ typical} + t_{WHGL} \times I_{CC4} \text{ typical})]$.
- Formula to calculate typical Erase/Erase Verify Power = $[V_{PP} (I_{PP3} \text{ typical} \times t_{ERASE} \text{ typical} + I_{PP5} \text{ typical} \times t_{WHGL} \times \# \text{ Bytes})] + [V_{CC} (I_{CC3} \text{ typical} \times t_{ERASE} \text{ typical} + I_{CC5} \text{ typical} \times t_{WHGL} \times \# \text{ Bytes})]$.
- One Complete Cycle = Array Preprogram + Array Erase + Program.
- "Typicals" are not guaranteed, but based on a limited number of samples from production lots.

ABSOLUTE MAXIMUM RATINGS*

Operating Temperature	
During Read0°C to +70°C(1)
During Erase/Program0°C to +70°C
Temperature Under Bias	
-10°C to +80°C
Storage Temperature	
-65°C to +125°C
Voltage on Any Pin with	
Respect to Ground-2.0V to +7.0V(2)
Voltage on Pin A ₉ with	
Respect to Ground-2.0V to +13.5V(2,3)
V _{PP} Supply Voltage with	
Respect to Ground	
During Erase/Program-2.0V to +14.0V(2,3)
V _{CC} Supply Voltage with	
Respect to Ground-2.0V to +7.0V(2)
Output Short Circuit Current	
100 mA(4)

NOTES:

1. Operating temperature is for commercial product defined by this specification.
2. Minimum DC input voltage is -0.5V. During transitions, inputs may undershoot to -2.0V for periods less than 20 ns. Maximum DC voltage on output pins is V_{CC} + 0.5V, which may overshoot to V_{CC} + 2.0V for periods less than 20 ns.
3. Maximum DC voltage on A₉ or V_{PP} may overshoot to +14.0V for periods less than 20 ns.
4. Output shorted for no more than one second. No more than one output shorted at a time.

OPERATING CONDITIONS

Symbol	Parameter	Limits		Unit	Comments
		Min	Max		
T _A	Operating Temperature	0	70	°C	For Read-Only and Read/Write Operations
V _{CC}	V _{CC} Supply Voltage	4.50	5.50	V	

NOTICE: This is a production data sheet. The specifications are subject to change without notice.

***WARNING:** Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.

Table 4. 28F256A Typical Update Power Dissipation

Operation	Power Dissipation (Watt-Seconds)	Notes
Array Program/Program Verify	0.0-1.0	1
Array Erase/Erase Verify	0.080	2
One Complete Cycle	0.100	3

DC CHARACTERISTICS—TTL/NMOS COMPATIBLE

Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typical(4)	Max		
I_{LI}	Input Leakage Current	1			± 1.0	μA	$V_{CC} = V_{CC\ max}$ $V_{IN} = V_{CC}$ or V_{SS}
I_{LO}	Output Leakage Current	1			± 10.0	μA	$V_{CC} = V_{CC\ max}$ $V_{OUT} = V_{CC}$ or V_{SS}
I_{CCS}	V_{CC} Standby Current	1		0.3	1.0	mA	$V_{CC} = V_{CC\ max}$ $CE\# = V_{IH}$
I_{CC1}	V_{CC} Active Read Current	1		10	30	mA	$V_{CC} = V_{CC\ max}$ $CE\# = V_{IL}$ $f = 6\ MHz$, $I_{OUT} = 0\ mA$
I_{CC2}	V_{CC} Programming Current	1, 2		1.0	10	mA	Programming in Progress
I_{CC3}	V_{CC} Erasure Current	1, 2		5.0	15	mA	Erasure in Progress
I_{CC4}	V_{CC} Program Verify Current	1, 2		5.0	15	mA	$V_{PP} = V_{PPH}$ Program Verify in Progress
I_{CC5}	V_{CC} Erase Verify Current	1, 2		5.0	15	mA	$V_{PP} = V_{PPH}$ Erase Verify in Progress
I_{PPS}	V_{PP} Leakage Current	1			± 10.0	μA	$V_{PP} \leq V_{CC}$
I_{PP1}	V_{PP} Read Current, ID Current, or Standby Current	1		90	200	μA	$V_{PP} > V_{CC}$
					± 10.0		$V_{PP} \leq V_{CC}$
I_{PP2}	V_{PP} Programming Current	1, 2		8.0	30	mA	$V_{PP} = V_{PPH}$ Programming in Progress
I_{PP3}	V_{PP} Erase Current	1, 2		4.0	20	mA	$V_{PP} = V_{PPH}$ Erasure in Progress
I_{PP4}	V_{PP} Program Verify Current	1, 2		2.0	5.0	mA	$V_{PP} = V_{PPH}$ Program Verify in Progress
I_{PP5}	V_{PP} Erase Verify Current	1, 2		2.0	5.0	mA	$V_{PP} = V_{PPH}$ Erase Verify in Progress
V_{IL}	Input Low Voltage		-0.5		0.8	V	
V_{IH}	Input High Voltage		2.0		$V_{CC} + 0.5$	V	
V_{OL}	Output Low Voltage				0.45	V	$I_{OL} = 5.8\ mA$ $V_{CC} = V_{CC\ min}$
V_{OH1}	Output High Voltage		2.4			V	$I_{OH} = -2.5\ mA$ $V_{CC} = V_{CC\ min}$
V_{ID}	A ₉ Intelligent Identifier Voltage		11.50		13.00	V	
I_{ID}	A ₉ Intelligent Identifier Current	1, 2		90	200	μA	$A_9 = V_{ID}$
V_{PPL}	V_{PP} During Read-Only Operations		0.00		6.5	V	Note: Erase/Program are Inhibited when $V_{PP} = V_{PPL}$
V_{PPH}	V_{PP} During Read/Write Operations		11.40		12.60	V	
V_{LKO}	V_{CC} Erase/Write Lock Voltage		2.5			V	

DC CHARACTERISTICS—CMOS COMPATIBLE

Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typical ⁽⁴⁾	Max		
I_{LI}	Input Leakage Current	1			± 1.0	μA	$V_{CC} = V_{CC\max}$ $V_{IN} = V_{CC}$ or V_{SS}
I_{LO}	Output Leakage Current	1			± 10.0	μA	$V_{CC} = V_{CC\max}$ $V_{OUT} = V_{CC}$ or V_{SS}
I_{CCS}	V_{CC} Standby Current	1		50	100	μA	$V_{CC} = V_{CC\max}$ $CE\# = V_{CC} \pm 0.2V$
I_{CC1}	V_{CC} Active Read Current	1		10	30	mA	$V_{CC} = V_{CC\max}$ $CE\# = V_{IL}$ $f = 6\text{ MHz}$, $I_{OUT} = 0\text{ mA}$
I_{CC2}	V_{CC} Programming Current	1, 2		1.0	10	mA	Programming in Progress
I_{CC3}	V_{CC} Erase Current	1, 2		5.0	15	mA	Erase in Progress
I_{CC4}	V_{CC} Program Verify Current	1, 2		5.0	15	mA	$V_{PP} = V_{PPH}$ Program Verify in Progress
I_{CC5}	V_{CC} Erase Verify Current	1, 2		5.0	15	mA	$V_{PP} = V_{PPH}$ Erase Verify in Progress
I_{PPS}	V_{PP} Leakage Current	1			± 10.0	μA	$V_{PP} \leq V_{CC}$
I_{PP1}	V_{PP} Read Current, I_D Current, or Standby Current	1		90	200	μA	$V_{PP} > V_{CC}$
					± 10.0	μA	$V_{PP} \leq V_{CC}$
I_{PP2}	V_{PP} Programming Current	1, 2		8.0	30	mA	$V_{PP} = V_{PPH}$ Programming in Progress
I_{PP3}	V_{PP} Erase Current	1, 2		4.0	20	mA	$V_{PP} = V_{PPH}$ Erase in Progress
I_{PP4}	V_{PP} Program Verify Current	1, 2		2.0	5.0	mA	$V_{PP} = V_{PPH}$ Program Verify in Progress
I_{PP5}	V_{PP} Erase Verify Current	1, 2		2.0	5.0	mA	$V_{PP} = V_{PPH}$ Erase Verify in Progress
V_{IL}	Input Low Voltage		-0.5		0.8	V	
V_{IH}	Input High Voltage		$0.7V_{CC}$		$V_{CC} + 0.5$	V	
V_{OL}	Output Low Voltage				0.45	V	$I_{OL} = 5.8\text{ mA}$ $V_{CC} = V_{CC\min}$
V_{OH1}	Output High Voltage		$0.85V_{CC}$			V	$I_{OH} = -2.5\text{ mA}$, $V_{CC} = V_{CC\min}$
V_{OH2}			$V_{CC} - 0.4$			V	$I_{OH} = 100\text{ }\mu A$, $V_{CC} = V_{CC\min}$
V_{ID}	A_9 Intelligent Identifier Voltage		11.50		13.00	V	
I_{ID}	A_9 Intelligent Identifier Current	1, 2		90	200	μA	$A_9 = V_{ID}$

DC CHARACTERISTICS—CMOS COMPATIBLE (Continued)

Symbol	Parameter	Notes	Limits			Unit	Test Conditions
			Min	Typical ⁽⁴⁾	Max		
V _{PPL}	V _{PP} During Read-Only Operations		0.00		6.5	V	Note: Erase/Program are Inhibited when V _{PP} = V _{PPL}
V _{PPH}	V _{PP} During Read/Write Operations		11.40		12.60	V	
V _{LKO}	V _{CC} Erase/Write Lock Voltage		2.5			V	

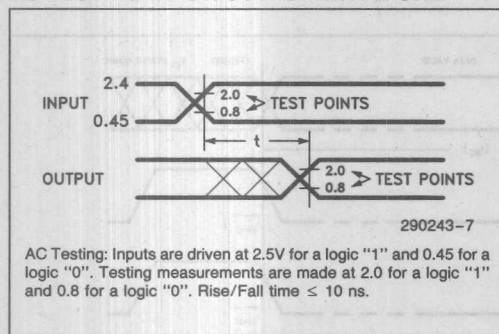
CAPACITANCE⁽³⁾ T = 25°C, f = 1.0 MHz

Symbol	Parameter	Notes	Limits		Unit	Conditions
			Min	Max		
C _{IN}	Address/Control Capacitance	3		6	pF	V _{IN} = 0V
C _{OUT}	Output Capacitance	3		12	pF	V _{OUT} = 0V

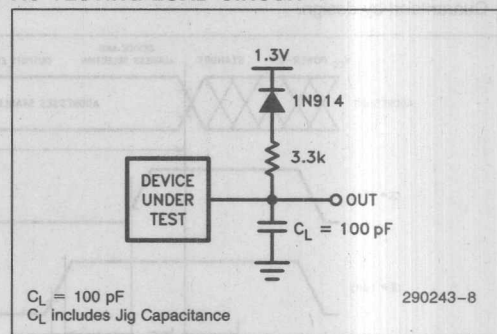
NOTES FOR DC CHARACTERISTICS AND CAPACITANCE:

1. All currents are in RMS unless otherwise noted. Typical values at V_{CC} = 5.0V, V_{PP} = 12.0V, T = 25°C. These currents are valid for all product versions (Packages and Speeds).
2. Not 100% tested: characterization data available.
3. Sampled, not 100% tested.
4. "Typicals" are not guaranteed, but based on a limited number of samples from production lots.

AC TESTING INPUT/OUTPUT WAVEFORM



AC TESTING LOAD CIRCUIT



AC Test Conditions

Input Rise and Fall Times (10% to 90%) 10 ns
Input Pulse Levels 0.45 and 2.4
Input Timing Reference Level 0.8 and 2.0
Output Timing Reference Level 0.8 and 2.0

AC CHARACTERISTICS Read-Only Operations

Versions		Notes	28F256A-120		28F256A-150		Unit
Symbol	Characteristic		Min	Max	Min	Max	
t_{AVAV}/t_{RC}	Read Cycle Time		120		150		ns
t_{ELQV}/t_{CE}	Chip Enable Access Time			120		150	ns
t_{AVQV}/t_{ACC}	Address Access Time			120		150	ns
t_{GLQV}/t_{OE}	Output Enable Access Time			50		55	ns
t_{ELQX}/t_{LZ}	Chip Enable to Output in Low Z	2, 3	0		0		ns
t_{EHQZ}	Chip Disable to Output in High Z	2		55		55	ns
t_{GLQX}/t_{OLZ}	Output Enable to Output in Low Z	2, 3	0		0		ns
t_{GHQZ}/t_{DF}	Output Disable to Output in High Z	2		30		35	ns
t_{OH}	Output Hold from Address, CE #, or OE # Change	1, 2	0		0		ns
t_{WHGL}	Write Recovery Time before Read		6		6		μ s

NOTES:

1. Whichever occurs first.
2. Sampled, not 100% tested.
3. Guaranteed by design.

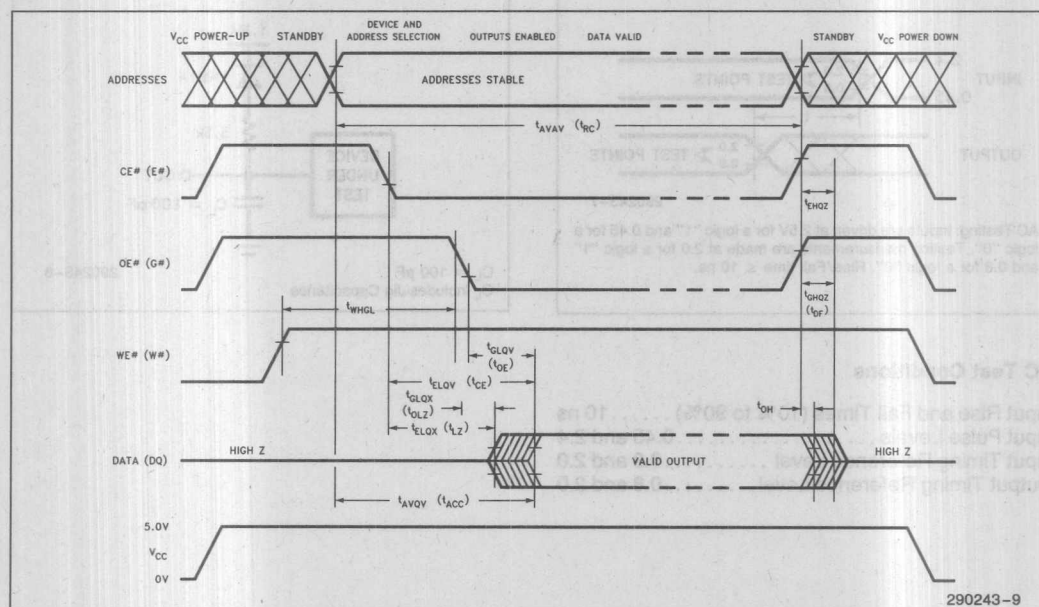


Figure 6. AC Waveform for Read Operations

AC CHARACTERISTICS—For Write/Erase/Program Operations(1)

Versions		Notes	28F256A-120		28F256A-150		Unit
Symbol	Characteristic		Min	Max	Min	Max	
t _{AVAV} /t _{WC}	Write Cycle Time		120		150		ns
t _{AVWL} /t _{AS}	Address Set-Up Time		0		0		ns
t _{WLAX} /t _{AH}	Address Hold Time		60		60		ns
t _{DVWH} /t _{DS}	Data Set-Up Time		50		50		ns
t _{WHDX} /t _{DH}	Data Hold Time		10		10		ns
t _{WHGL}	Write Recovery Time before Read		6		6		μs
t _{GHWL}	Read Recovery Time before Write	2	0		0		μs
t _{ELWL} /t _{CS}	Chip Enable Set-Up Time before Write		20		20		ns
t _{WHEH} /t _{CH}	Chip Enable Hold Time		0		0		ns
t _{WLWH} /t _{WP}	Write Pulse Width		60		60		ns
t _{WHWL} /t _{WPH}	Write Pulse Width High		20		20		ns
t _{WHWH1}	Duration of Programming Operation	3	10		10		μs
t _{WHWH2}	Duration of Erase Operation	3	9.5		9.5		ms
t _{VPEL}	V _{PP} Set-Up Time to Chip Enable Low	2	1.0		1.0		μs

NOTES:

1. Read timing parameters during read/write operations are the same as during read-only operations. Refer to AC Characteristics for Read-Only Operations.
2. Guaranteed by design.
3. The integrated stop timer terminates the programming/erase operations, thereby eliminating the need for a maximum specification.

ERASE AND PROGRAMMING PERFORMANCE

Parameter	Notes	Limits						Unit
		28F256A-120			28F256A-150			
		Min	Typ	Max	Min	Typ	Max	
Chip Erase Time	1, 3, 4		1	10		1	10	sec
Chip Program Time	1, 2, 4		0.5	3		0.5	3	sec

NOTES:

1. "Typicals" are not guaranteed, but based on a limited number of samples taken from production lots. Data taken at 25°C, 12.0V V_{PP}, at 0 cycles.
2. Minimum byte programming time excluding system overhead is 16 μs program + 6 μs write recovery), while maximum is 400 μs/byte (16 μs x 25 loops allowed by algorithm). Max chip programming is specified lower than the worst case allowed by the programming algorithm since most bytes program significantly faster than the worst case byte.
3. Excludes 00H Programming Prior to Erasure.
4. Excludes System-Level Overhead.
5. Refer to RR-60 "ETOX" II Flash Memory Reliability Data Summary for typical cycling data and failure rate calculations.

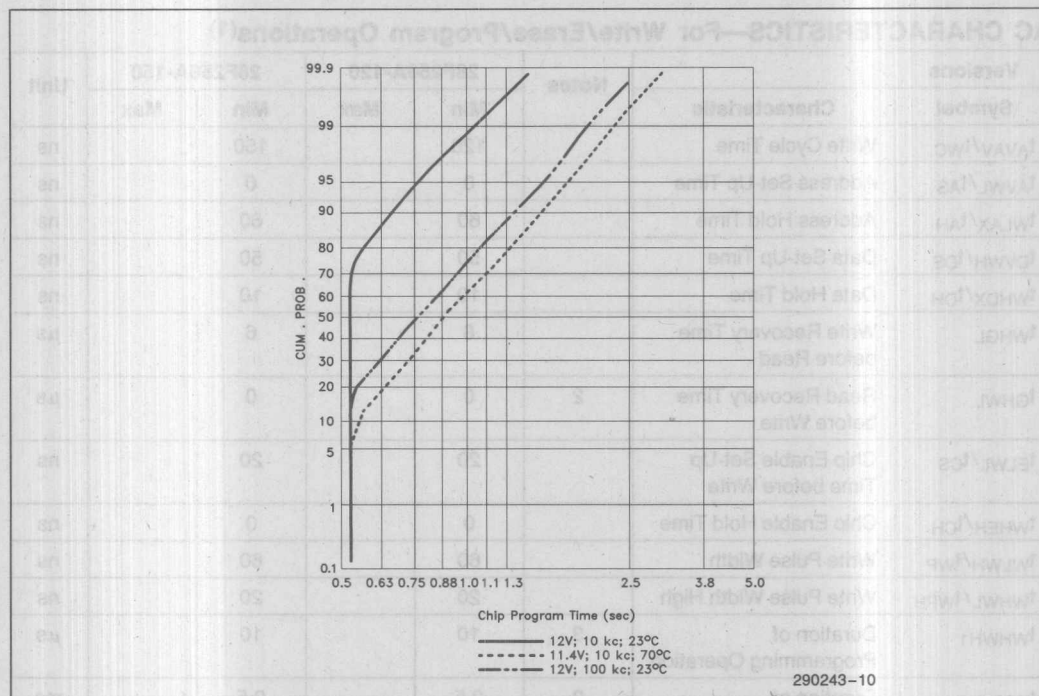


Figure 7. 28F256A Typical Programming Capability

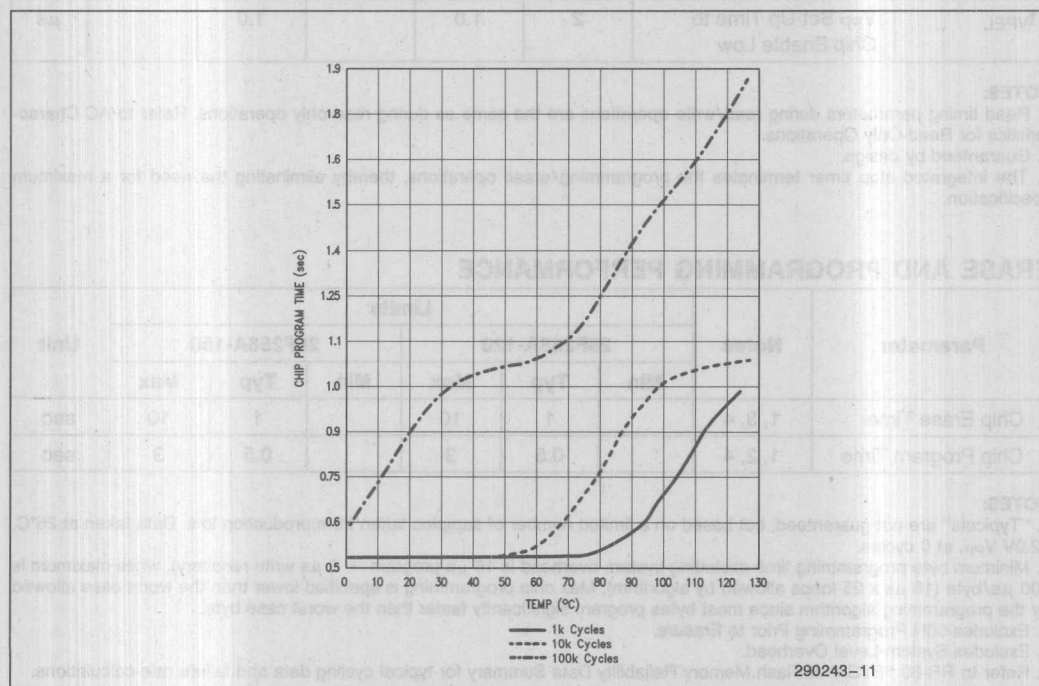
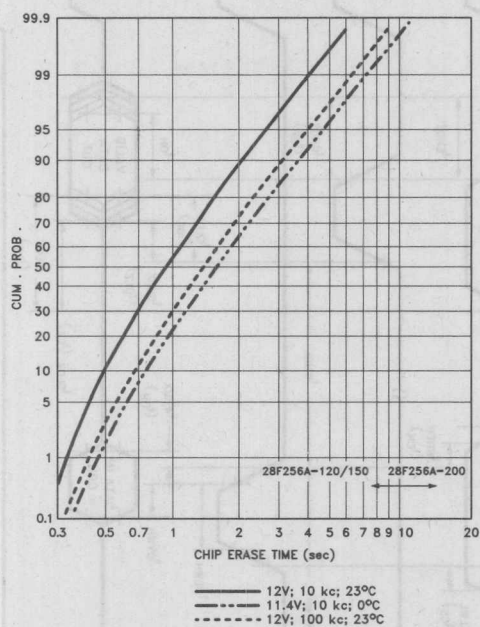
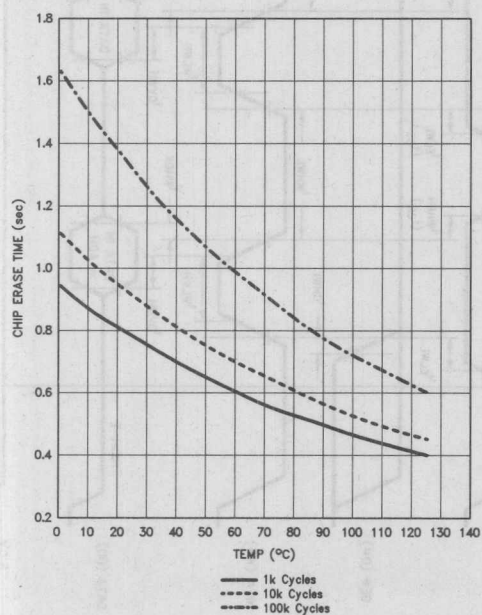


Figure 8. 28F256A Typical Program Time at 12V



290243-12

Figure 9. 28F256A Typical Erase Capability



290243-13

Figure 10. 28F256A Typical Erase Time at 12.0V

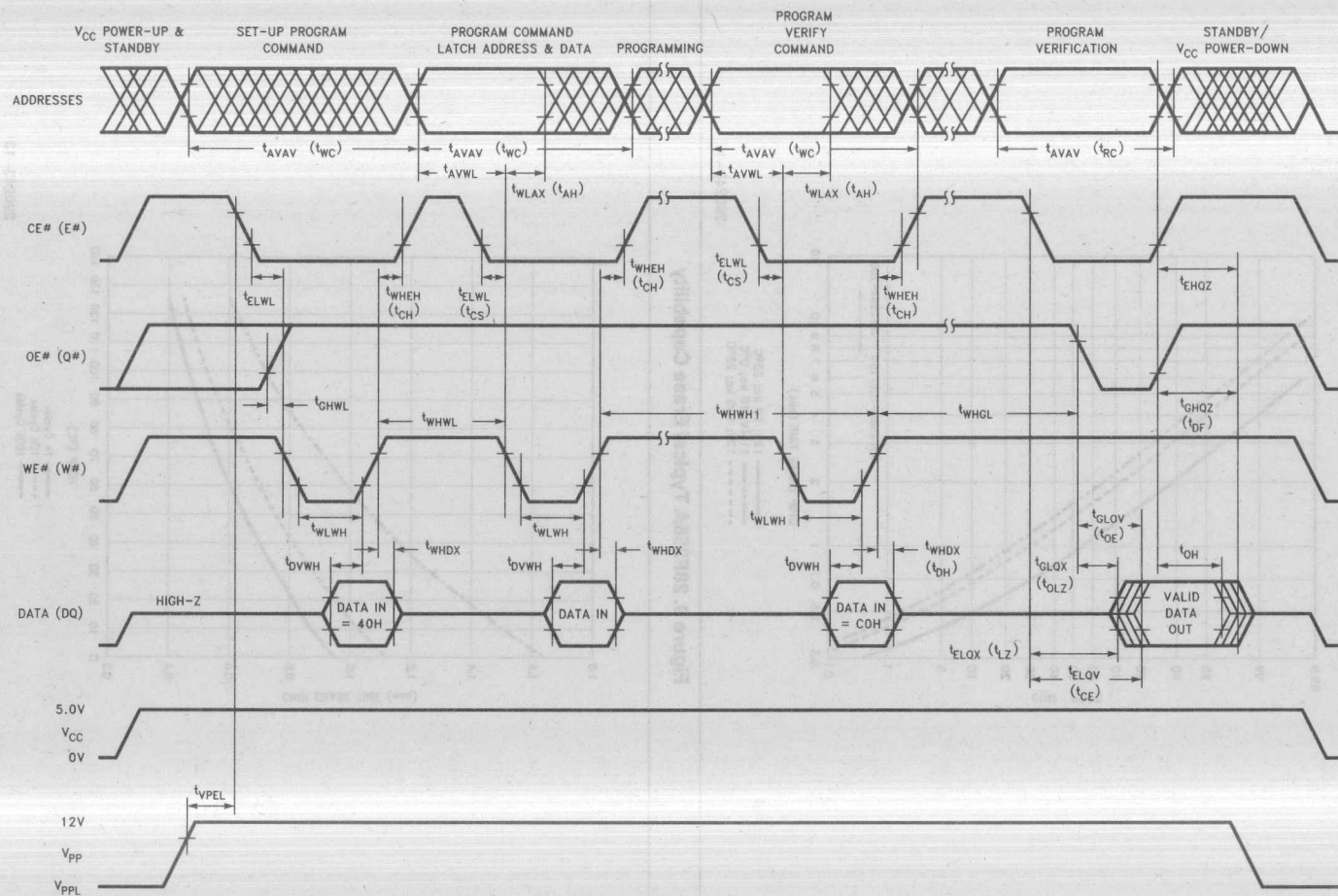


Figure 11. AC Waveforms for Programming Operations

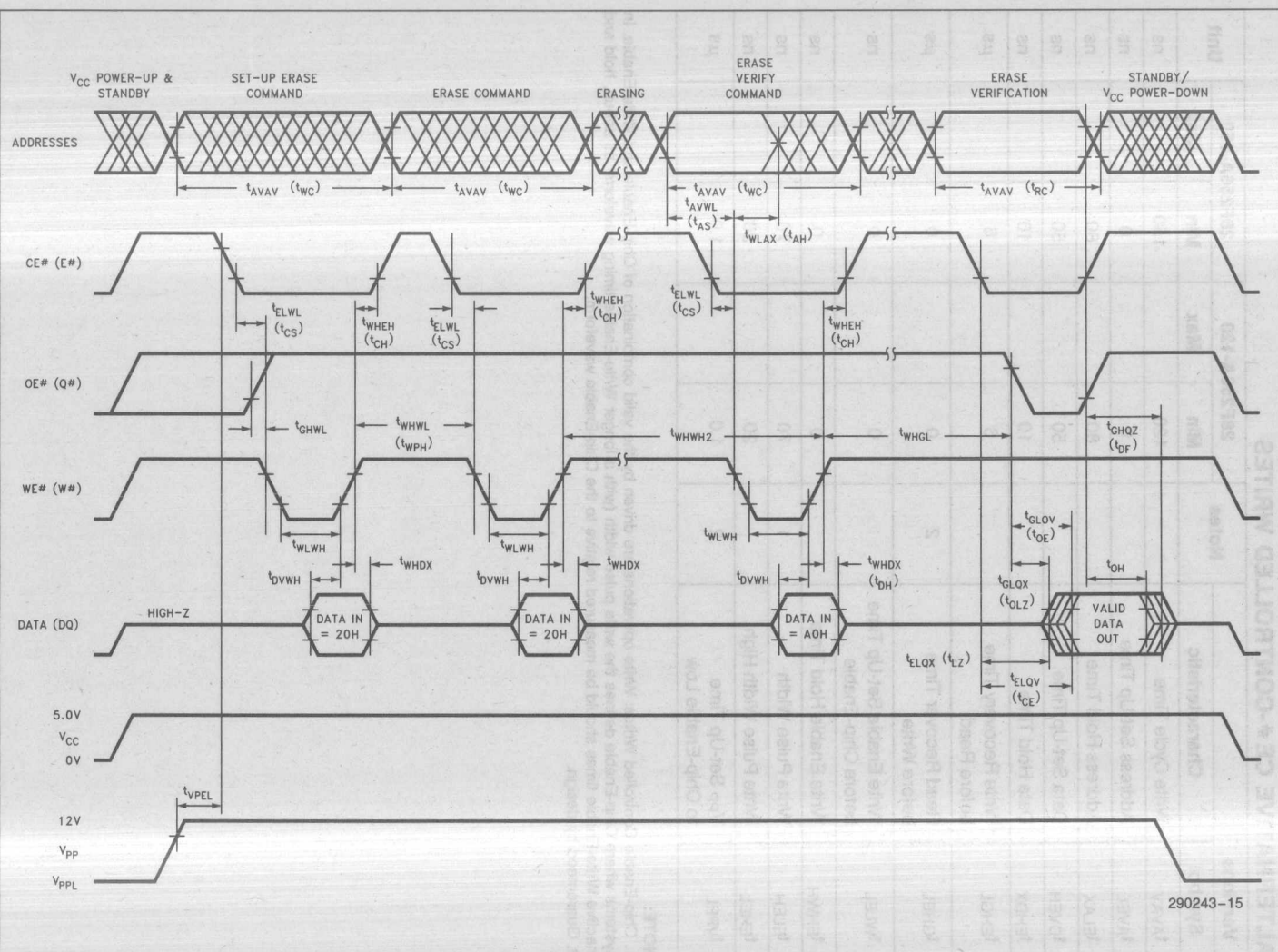


Figure 12. AC Waveforms for Erase Operations

ALTERNATIVE CE #-CONTROLLED WRITES

Versions		Notes	28F256A-120		28F256A-150		Unit
Symbol	Characteristic		Min	Max	Min	Max	
t_{AVAV}	Write Cycle Time		120		150		ns
t_{AVEL}	Address Set-Up Time		0		0		ns
t_{ELAX}	Address Hold Time		80		80		ns
t_{DVEH}	Data Set-Up Time		50		50		ns
t_{EHDX}	Data Hold Time		10		10		ns
t_{EHGL}	Write Recovery Time before Read		6		6		μs
t_{GHLE}	Read Recover Time before Write	2	0		0		μs
t_{WLEL}	Write Enable Set-Up Time before Chip-Enable		0		0		ns
t_{EHWH}	Write Enable Hold Time		0		0		ns
t_{ELEH}	Write Pulse Width	1	70		70		ns
t_{EHEL}	Write Pulse Width High		20		20		ns
t_{VPEL}	V_{pp} Set-Up Time to Chip-Enable Low	2	1.0		1.0		μs

NOTE:

1. Chip-Enable Controlled Writes: Write operations are driven by the valid combination of Chip-Enable and Write-Enable. In systems where Chip-Enable defines the write pulse width (with a longer Write-Enable timing waveform) all set-up, hold and inactive Write-Enable times should be measured relative to the Chip-Enable waveform.

2. Guaranteed by design.



Figure 13. Alternate AC Waveforms for Programming Operations

ORDERING INFORMATION

<div> <div>P28F256A-120</div> <div>P28F256A-150</div> </div>										
PACKAGE						ACCESS SPEED (ns)				
P = 32-PIN PLASTIC DIP						120 ns				
N = 32-LEAD PLCC						150 ns				
VALID COMBINATIONS:						290243-17				
P28F256A-120						N28F256A-120				
P28F256A-150						N28F256A-150				

ADDITIONAL INFORMATION

ER-20,	"ETOX II Flash Memory Technology"	Order Number
ER-24,	"Intel Flash Memory"	294005
RR-60,	"ETOX II Flash Memory Reliability Data Summary"	294008
AP-316,	"Using Flash Memory for In-System Reprogramming Nonvolatile Storage"	293002
AP-325,	"Guide to Flash Memory Reprogramming"	292046
		292059

REVISION HISTORY

Number	Description
004	Removed Preliminary Classification. Removed 200 ns speed bin. Revised Erase Maximum Pulse Count for Figure 5 from 3000 to 1000 . Clarified AC and DC test conditions.
005	Corrected AC waveforms.
006	Revised symbols; i.e., \overline{CE} , \overline{OE} , etc. to CE#, OE#, etc.

Using Flash Memory for In-System Reprogrammable Nonvolatile Storage

5

SAUL ZALES
DALE ELBERT
APPLICATIONS ENGINEERING
INTEL CORPORATION

September 1993

USING FLASH MEMORY FOR IN-SYSTEM REPROGRAMMABLE NONVOLATILE STORAGE

CONTENTS	PAGE
1.0 INTRODUCTION	5-120
1.1 PROM Programmer vs System- Processor Controlled Programming	5-120
1.2 Information Download and Upload	5-120
Version Updates (Download)	5-120
Data Acquisition (Upload)	5-120
2.0 DEVICE FEATURES AND ISW APPLICATION CONSIDERATIONS	5-121
2.1 Flash Memory Pinouts	5-121
2.2 Command Register Architecture	5-123
Simplified Processor Interface ..	5-123
Command Register Reset	5-124
Data Protection on Power Transitions	5-124
2.3 V _{PP} Specifications	5-125
3.0 HARDWARE DESIGN FOR ISW ...	5-125
3.1 V _{PP} Generation	5-125
3.1.1 Regulating Down from Higher Voltage	5-125
3.1.2 Pumping 5V up to 12V	5-125
3.1.3 Absolute Data Protection — V _{PP} On/Off Control	5-126
3.1.4 Writes and Reads during V _{PP} Transitions	5-126
3.1.5 Other V _{PP} Considerations	5-126
3.1.6 V _{PP} Circuitry and Trace Layout	5-127
3.2 Communications — Getting Data to and from the Flash Memory	5-127
4.0 SOFTWARE DESIGN FOR ISW	5-127
4.1 System Integration — Boot Code Requirements	5-127
4.1.1 ISW Flag Check	5-128
4.2 Communication Protocols and Flash Memory ISW	5-128
4.3 Data Accumulation Software Techniques	5-129

CONTENTS

PAGE

4.4 Reprogramming Routines	5-129
4.4.1 Quick-Erase Algorithm	5-129
Algorithm Timing Delays ..	5-129
High Performance Parallel Device Erasure	5-130
4.4.2 Quick-Pulse Programming	
Algorithm	5-131
Algorithm Timing Delays ..	5-131
High Performance Parallel Device Programming ...	5-131
4.4.3 Pulse Width Timing Techniques	5-132
Software Methods and Examples	5-132
Hardware Methods	5-132
5.0 SYSTEM DESIGN EXAMPLE: AN 80C186 DESIGN	5-133
6.0 SUMMARY	5-134

CONTENTS

PAGE

APPENDIX A: ON BOARD PROGRAMMING DESIGN CONSIDERATIONS	5-135
APPENDIX B: V _{PP} GENERATION CIRCUITS	5-145
APPENDIX C: LIST OF DC-DC CONVERTER COMPANIES	5-149
APPENDIX D: DETAILED PARALLEL ERASE FLOW CHART	5-150
APPENDIX E: DETAILED PARALLEL PROGRAMMING FLOW CHART	5-152
APPENDIX F: DETAILED SYSTEM SCHEMATICS ...	5-154

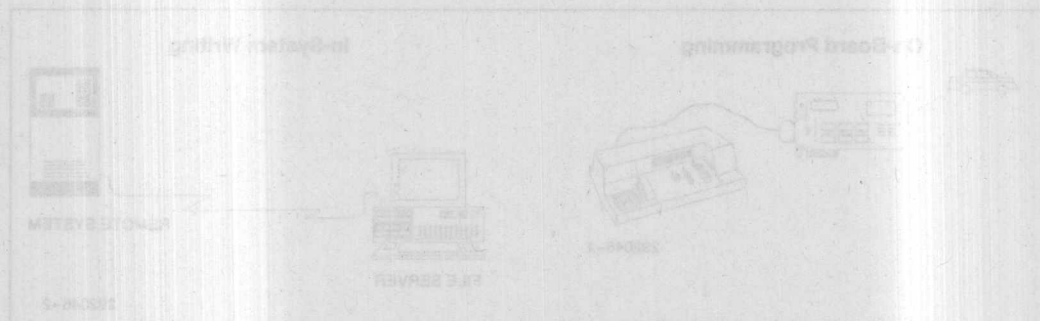


Figure 1. Two diagrams illustrate OBP and ISW. In OBP, a PROM programmer updates a system's flash memory. The ISW diagram shows a host updating remote flash memory via serial link. The remote system performs the flash reprogramming with its own CPU.

1.0 INTRODUCTION

Intel's ETOX II (EPROM tunnel oxide) flash memory technology uses a single-transistor cell to provide in-system reprogrammable nonvolatile storage. Reprogramming entails electrically erasing all bits in parallel and then randomly programming any byte in the array. This new technology offers designers alternatives for two of industry's needs: 1) a cost-effective means of updating program code; and 2) a solid-state approach for non-volatile data accumulation or storage.

This application note:

- introduces you to the concepts of in-system writing;
- discusses the hardware and software considerations for reprogramming flash memories in-system;
- offers a checklist for integrating Intel's flash memories into microprocessor- or microcontroller-based systems; and
- shows an example of an 80C186 design which incorporates flash memory.

1.1 PROM Programmer vs System-Processor Controlled Programming

While soldered to a printed circuit board, one of two sources controls flash memory reprogramming: 1) a PROM programmer connected to the board, or 2) the system's own central processing unit (CPU). These are called on-board programming (OBP), and in-system writing (ISW), respectively. With OBP, the PROM programmer supplies the programming voltage (V_{pp}) and the programming intelligence; with ISW, V_{pp} is generated locally and the system itself drives the reprogramming process. Both methods offer a variety of benefits. However this application note focuses on ISW.

NOTE:

See Appendix A for OPB design considerations.

1.2 Information Download and Upload

ETOX II flash memory technology programs extremely quick, permitting "on-the-fly" programming with unbuffered 19.2K baud data input. The remote ISW system handles the serial communication link for the host interface, as well as the flash memory reprogramming.

Version Updates (Download)

Flash memories enable code version updates using simple hardware designs. Beyond the basic system, a local V_{pp} supply is all that is needed for remote code download.

A central host computer can download program code to many remote systems. Flash memory offers this capability without the drawbacks of other technologies. It is solid-state and nonvolatile, thus eliminating mechanical component wear-out (common with disk drives) and the risk of losing updates (a concern with battery-backed RAM). These aspects of flash memory offer major advantages in automated factories, remote systems, portable equipment and other applications. Finally, flash memories provide this capability at a much lower cost than byte-alterable EEPROM and battery-backed SRAM.

Data Acquisition (Upload)

Intel's flash memories allow single-byte programming for data accumulation applications. A remote data-logger uploads its information to a central host via serial link. The flash memory device is then in-system erased

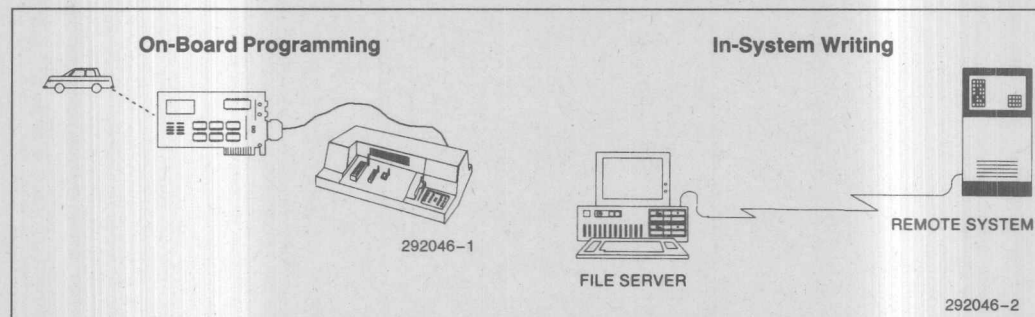


Figure 1. These diagrams illustrate OBP and ISW. In OBP, a PROM programmer updates a system's flash memory. The ISW diagram shows a host updating remote flash memory via serial link. The remote system performs the flash reprogramming with its own CPU.

... power meter, for example, it could be configured to track and monitor power usage and report the data to a central computer for billing and utility management. This reduces the cost of manual door-to-door meter reading.

2.0 DEVICE FEATURES AND ISW APPLICATION CONSIDERATIONS

This section gives a brief overview of Intel's flash memory features and explains how they facilitate ISW design.

The 32-pin DIP memory site is forward-compatible from the 256K bit to the 2 Mbit flash memory density. It fits into the 27C010 Mbit EPROM pinout and requires no multiplexed pins. Also, with just a single circuit-board jumper trace, a 28-pin EPROM can be placed in the lower pins of the 32-pin flash memory site. (See Figures 2A and 2B, Flash Memory Pinouts.) For more information on intertechnology pin compatibility see Ap Brief AB-25.

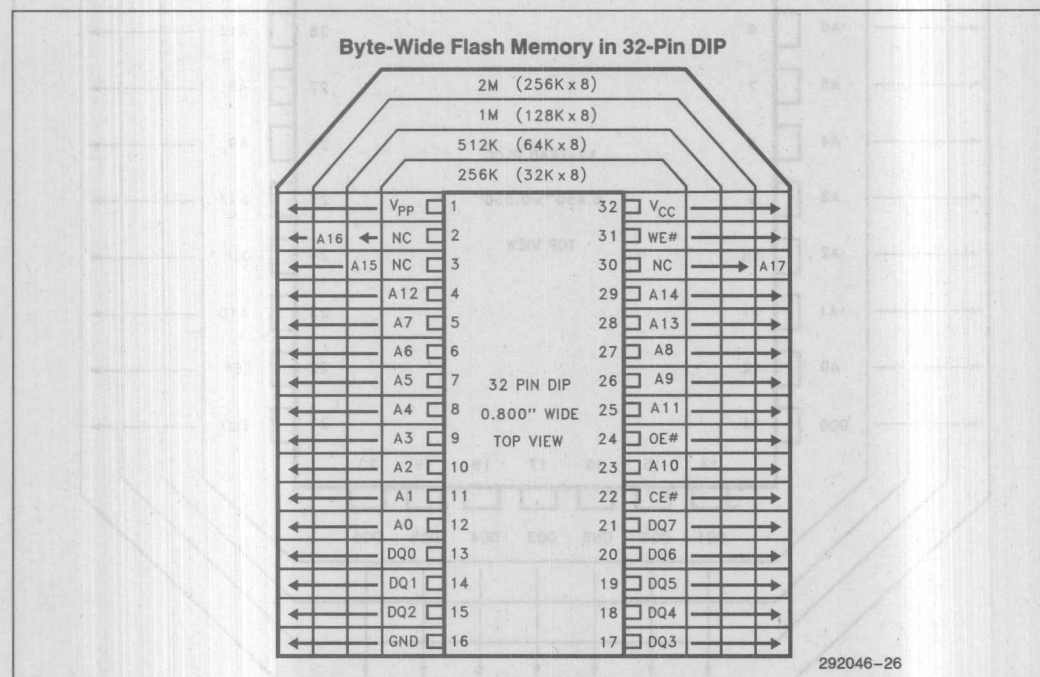


Figure 2A. Flash Memory Pinouts

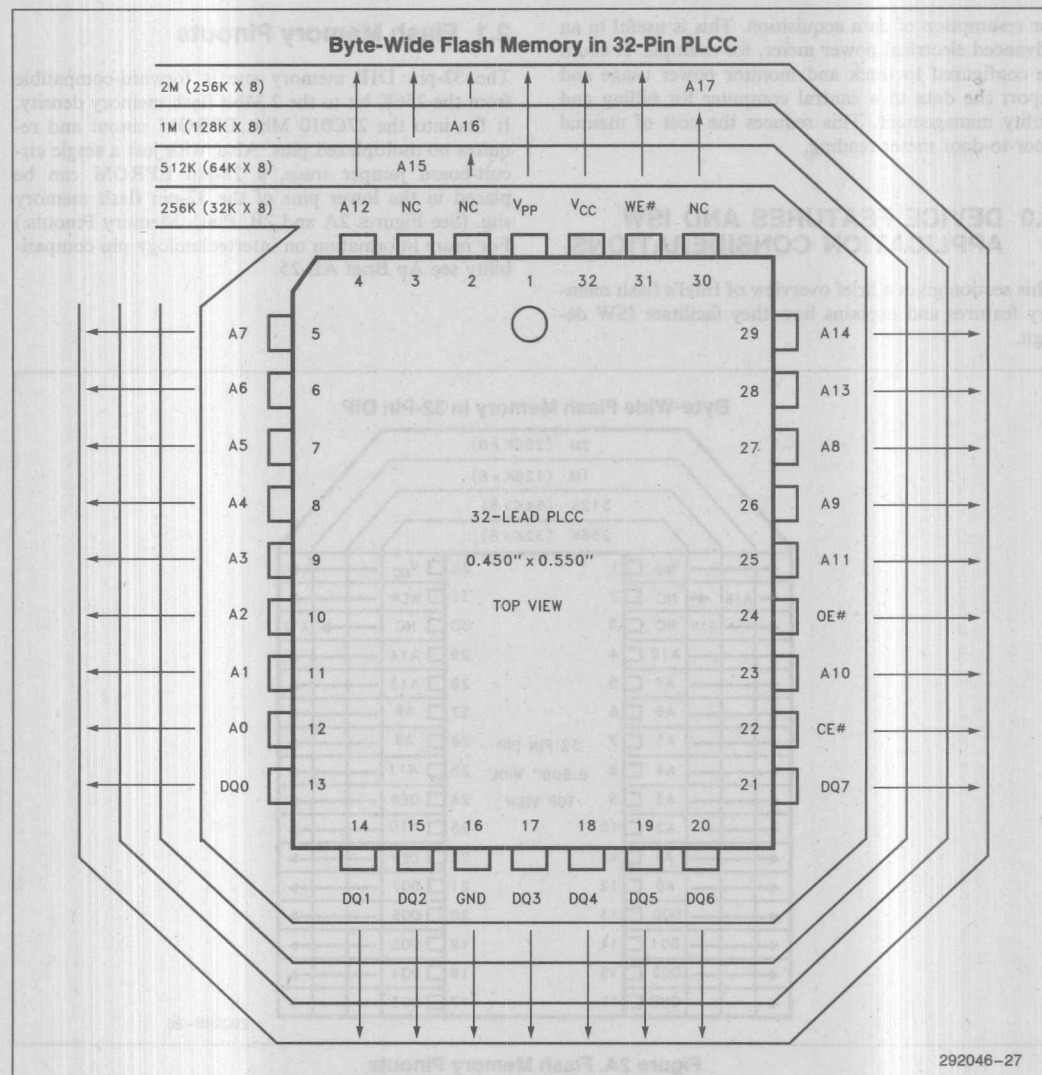


Figure 2B. Flash Memory Pinouts

Table 1. Command Register Instructions

Command	Bus Cycles Req'd	First Bus Cycle			Second Bus Cycle		
		Operation	Addr(1)	Data(2)	Operation	Addr(1)	Data(2)
Read Memory ⁽³⁾	1	Write	X	00H	Read	Valid	Valid
Read Intelligent Identifier	1	Write	X	90H	Read	00/01H	ID
Set-Up Erase/Erase	2	Write	X	20H	Write	X	20H
Erase Verify	2	Write	EA	A0H	Read	X	EVD
Set-Up Program/Program	2	Write	X	40H	Write	PA	PD
Program Verify	2	Write	X	C0H	Read	X	PVD
Reset ⁽³⁾	2	Write	X	FFH	Write	X	FFH

NOTES:

1. Addresses are latched on the falling edge of the Write-Enable pulse.

EA = Address of memory location to be read during erase verify.

PA = Address of memory location to be read during program verify.

2. EVD = Data read from location EA during erase verify.

PD = Data to be programmed at location PA. Data is latched on the rising edge of Write-Enable.

PVD = Data read from location PA during program verify. PA is latched on the Program command.

3. The second bus cycle must be followed by the next desired command register write, given the proper delay times.

2.2 Command Register Architecture

Simplified Processor Interface

Intel's command register architecture simplifies the processor interface. The command register allows CE#, WE#, and OE# to have standard read/write functionality. All commands such as "Set-up Program" or "Program Verify" can be written with standard system timings. Raising V_{pp} to 12V enables the command register for memory read/write operation, while lowering V_{pp} below V_{CC} + 2V restores the device to a read only memory.

Writing to the register toggles an internal state-machine. The state-machine output controls device functionality. Some commands require one write cycle, while others require two. The command register itself does not occupy an addressable memory location. The register simply stores the command, along with address and data needed to execute the command. With this architecture, the device expects the first write cycle to be a command and does not corrupt data at the specified address. Table 1 contains a list of command register instructions.

The following sections describe the commands in relation to device operation. For more information on the command register see the appropriate flash memory data sheets, and Section 4.4 "Reprogramming Routines".

Read Memory Command—00H

This command allows for normal memory read operations with V_{pp} turned on. After writing the command and waiting 6 μ s, the CPU can read from the memory

at system speeds. Once placed in the read mode no further action is required on the command register for subsequent read operations.

Read Intelligent Identifier Command—90H

Most PROM programmers read the device's intelligent identifier to select the proper programming algorithm. On EPROMs, raising A9 to the V_{pp} level configures the device for this purpose. Since this is unacceptable in-system, you can read the flash memory intelligent identifier by first writing command 90H. Follow this by reading address 0000 and 0001H for the manufacturer and device ID. Reset the device with the Read Memory command after you have read the identifier.

Set-Up Erase/Erase Commands—20H

Write this command (20H) twice in succession to initiate erasure. The first write cycle sets up the device for erasure. The device starts erasing itself on the second command's rising edge of Write-Enable. Erasure is stopped when the CPU issues the Erase Verify command or when the device's integrated stop timer times out. Integrated stop timers provide a safety net for complex system environments. In these environments, s/w timer accuracy may be difficult to achieve. Some method of timing is still required, however the timer need only meet a minimum specification (10 ms). This is far easier than calibrating a timer to meet both a minimum and maximum specification (10 ms \pm 500 μ s).

NOTE:

Prior to erasure, it is necessary to program all bytes to the same level (data = 00H). See the Quick-Erase algorithm for more details.

Erase Verify Command—A0H

The erase command erases all bytes of the array in parallel. After each erase operation, all bytes must be verified to see if they are erased. Write the Erase Verify command (A0H) to stop erasure and setup verification.

Alternatively, you may allow the internal stop timer to halt erasure. You must still issue the Erase Verify command to set up verification.

The device latches the address to be verified on the falling edge of WE# and the actual command on the rising edge. Wait 6 μ s before reading the data at the address specified on the previous write cycle.

The flash memory applies an internally-generated reference voltage to the addressed byte. Reading 0FFH from the addressed byte in this mode indicates that all bits in the byte are erased with sufficient margin to V_{CC} and temperature fluctuations.

If the location is erased, then repeat the Erase Verify procedure for the next address location. Write the command prior to each byte verification to latch the byte's address. Continue this process for each byte in the array until a byte does not return 0FFH data, or the last address is accessed.

In the case where the data read is not 0FFH, perform another erase operation. (Refer to Set-up Erase/Erasure). Continue verifying from the address of the last verified byte. Once you have accessed the last address, erasure is complete and you can proceed to program the device. Terminate the erase verify operation by writing another valid command (e.g., Program Set-up).

Set-up Program/Program Commands—40H

Write this command (40H) twice in succession to initiate programming. The first write cycle sets up the device for programming. The device latches address and data on the falling and rising edges of the second write cycle, respectively. It also begins programming on the rising edge. You stop the programming operation by issuing the Program Verify command, or by allowing the integrated program stop timer to time out. This timer works similar to the erase stop timer. Again, a minimum specification replaces a tougher minimum/maximum combination (10 μ s–25 μ s).

Program Verify Command—C0H

Flash memory devices program on a byte-by-byte basis. After each programming operation, the byte just programmed must be verified. Write the Program Verify command (C0H) to stop programming and set-up verification. Should your software allow the integrated stop timer to halt programming, the software must resume the algorithm with the Program Verify command. The

device executes this command on the rising edge of Write-Enable. The program Verify command stages the device for verification of the byte last programmed. No new address information is latched.

The flash memory applies an internally-generated reference voltage to the addressed byte. Wait 6 μ s for the internal voltages to settle before reading the data at the address programmed. Reading valid data indicates that the byte programmed successfully.

Command Register Reset—FFH

Flash memories reset to the read mode during power-up, and remain in this mode as long as V_{PP} is less than $V_{CC} + 2V$. If your system leaves V_{PP} turned-on during a *system reset*, then incorporate a command register *device reset* into the hardware initialization routines. This is necessary because the CPU might be controlling programming or erasure when the system reset hits.

Write the reset command (0FFH) twice in succession to reset the device. The double write is necessary because of the state-machine reprogramming structure. For example, suppose the *system* is reset after a Set-up Program command. The flash memory state machine expects the next write cycle to contain valid address and data for programming, followed by another write cycle for program verification. The first Reset command will be mistaken for program data but will not corrupt the existing data. This is because the command (data = 0FFH) is a null condition for flash memory programming. Only data bits programmed to zero pull charge onto the memory cell and change the data. The second write cycle actually resets the device to the read function. Following the second reset cycle, you can write the next command (Read, Program Set-up, Erase Set-up, etc.).

If the V_{PP} supply is turned off upon system reset, the software reset is not required. The flash memory will reset itself automatically when V_{PP} powers down.

Data Protection on Power Transitions

The command register architecture offers another benefit in addition to simplified processor interface—during system power-up and power-down it protects data from corruption by unstable logic. Erasure or programming require V_{PP} to be greater than $V_{CC} + 2V$ and the proper command sequence to be initiated. For example the CPU must write the erase command twice in succession. The odds of this occurring randomly are slim. Additionally, should V_{PP} ramp to 12V prior to V_{CC} ramping past 2.5V, the device will lock out all spurious writes and internally block 12V from the flash memory cells. For even greater security, you can switch V_{PP} as discussed in Section 3.13.

2.3 Vpp Specifications

Flash memories, like EPROMs, require a 12V externally-generated power supply for reprogramming. Intel's Vpp specifications 12.0V \pm 0.6V (5%) is compatible with most off-the-shelf (or available in-system) power supplies. (Note, Section 3.1 discusses Vpp generation techniques, and Appendix B shows different circuit alternatives.)

It is essential to use the specified Vpp when reprogramming the flash memory device. Once the command to erase, program, or verify is issued, the device internally derives the required voltages from the Vpp supply. The command register controls selection of internal reference circuitry tapped off of Vpp. An improper Vpp level causes the references to be wrong, degrading the performance of the part.

(When programming U.V. EPROMs, VCC is raised to 6.5V. On flash memories, the Vpp reference circuitry and command register architecture provide the same function while keeping VCC and Vpp at static levels. An incorrect VCC level during U.V. EPROM programming poses similar hazards to improper Vpp levels on flash memories.)

The hardware design section discusses various methods for generating Vpp.

3.0 HARDWARE DESIGN FOR ISW

Covered in this section are the following:

- Description of ISW-specific functional system blocks including memory requirements
- Vpp generation techniques
- Communication Considerations

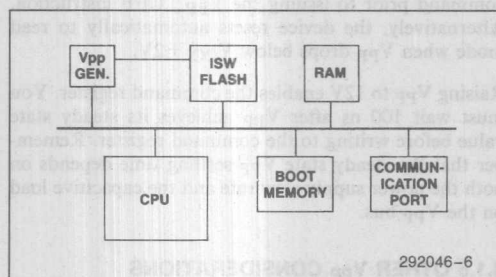


Figure 3. System Block Diagram

System Level Hardware Requirements for ISW:

- processor or controller
- limited amount EP/ROM or other flash memory devices for boot code, communications s/w, and reprogramming algorithms

- limited amount of RAM for variable storage (i.e., stacks, buffers, and other changing parameters)
- data import capability (i.e., serial line, LAN, floppy disk)
- flash memory for nonvolatile code or data storage needs
- Vpp generator or regulator

All of the functional blocks in Figure 3 are typical of any embedded or reprogrammable system with the exception of the Vpp generator. Some microcontrollers have on-chip EP/ROM, RAM and a serial port. With these devices, implementation of the ISW capability requires little additional hardware.

The next section discusses Vpp generation techniques and communications design considerations.

3.1 Vpp Generation

A static Vpp is needed to reprogram flash memories. The Vpp voltage can be generated by:

- 1) regulating it down from a higher voltage;
- 2) pumping it up from a lower voltage (i.e., charge pump, DC/DC converter, etc.); or
- 3) designing or specifying the system's 12V supply with the required ISW tolerances and specifications.

Sufficient current for reprogramming should be considered when selecting your Vpp generation option. Parallel reprogramming for flash memory in 16-bit or 32-bit systems will require, respectively, 2X or 4X additional current capability.

3.1.1 REGULATING DOWN FROM HIGHER VOLTAGE

Vpp is obtained from a higher voltage by using a linear regulator. Given the higher voltage, regulation offers the least expensive method of generating Vpp. Standard three terminal 12V \pm 1%, \pm 2%, \pm 4% non-adjustable regulators are available off-the-shelf. Some regulators have on/off control built-in. (See Appendix B, Vpp Circuit #1.) All regulators require a minimum input voltage greater than the output voltage. (See Appendix B, Vpp Circuit #2 and #3.)

3.1.2 PUMPING 5V UP TO 12V

Vpp can be obtained by pumping VCC and regulating it to the proper voltage. A voltage charge-pump can be designed and built by using a charge-pump integrated circuit and some discrete components (see Appendix B, Vpp Circuit #4) or by using a monolithic DC/DC converter (see Appendix A, Vpp Circuit #5).

When using adjustable circuits containing discrete components, design the output voltage so it falls within the Vpp specifications for all corners of the components'

skew (i.e., $V_{CC} \pm 10\%$; $R_x \pm 1\%$, $R_y \pm 1\%$, etc.). Include the resistors' temperature coefficients in the calculation matrix. Note that each of the various components can add error to the V_{pp} supply.

The monolithic DC/DC converter shown in Appendix B Circuit #5 fits into a 24-pin socket. It offers the advantages of close temperature tracking and ease of implementation. It has also been characterized at temperatures and meets all the V_{pp} specifications. Appendix C contains a partial list of vendors selling DC/DC converters.

Most DC/DC converters are only 50–60% efficient, so heat dissipation may be a concern. Some discrete boost circuits such as Appendix B, Circuit #4, offer much higher efficiency (70–85%). Circuit #4 as shown can supply 200 mA. Smaller inductor and capacitor component values and higher frequency boost converters can be used where less power is required. For example, designs which reprogram one or two flash memories simultaneously might use the LT1172. (Contact Linear Technologies for more information.)

In all V_{pp} generation methods, a capacitor on the input voltage terminals reduces the output noise voltage. Some power supplies (Appendix B, Circuits #3 and #4) specify a large-valued capacitor to decrease the Effective Series Resistance (ESR). Place a 0.1 μF capacitor within 0.25 inches of each flash memory's V_{pp} input (in addition to the one on the V_{pp} generator's input).

NOTE:

The ESR is inversely proportional to the capacitance value and the rated working voltage. To lower the ESR choose a capacitor with a large capacitance and a high working voltage (i.e., above 100V).

3.1.3 ABSOLUTE DATA PROTECTION— V_{pp} ON/OFF CONTROL

With V_{pp} below $V_{CC} + 2\text{V}$ or V_{CC} below 2.5V, internal circuitry disables the command register and eliminates the possibility of inadvertent erasure or programming. Switching the V_{pp} supply off provides the secondary benefits of improved power and thermal management.

There are two ways to switch V_{pp} on and off:

- 1) directly switch the V_{pp} generator's output, or
- 2) switch the input voltage supplying the regulation circuit.

Any switching circuit will cause a voltage drop, so choose a switch with this drop in mind. Some power supplies have asymmetrical tolerances on 12V (i.e., +5%, –4%). Flash memory allows the 12V supply to drop as low as –5%. The 1% difference between the supply and the device requirement allows the switch to have an ON resistance voltage drop of 0.12V. Continuing with this example, assume the system only reprograms one flash memory at a time. The current through

the switch into the flash is $I_{pp} = 30\text{ mA}$. Solving for the allowable resistance across the switch: $R = V/I = (0.12\text{V})/(30\text{ mA}) = 4\text{ Ohms}$. See Figure 4. Example Voltage Drop Across Switch. Note, one can reduce the effective $R_{DS}(\text{ON})$ by placing 2 or more FETs in parallel if necessary.

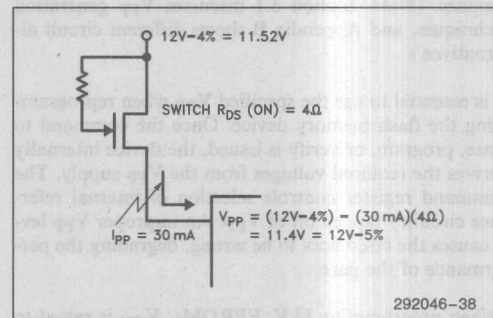


Figure 4

Controlling the input voltage of a DC/DC converter with a MOSPOWER FET is another straightforward approach. (See Appendix B, Circuit #5.) Choose the FET switch carefully. It should have a very low on-resistance to minimize the voltage divider effect of the converter and FET switch. If the voltage across the FET switch is too high, the converter will not have the proper input voltage to meet its specifications. Always design the switching circuit with sufficient margin to maximum V_{pp} and V_{CC} load currents.

3.1.4 WRITES AND READS DURING V_{pp} TRANSITIONS

After switching V_{pp} off, the CPU can read from the flash memory without waiting for the capacitors on V_{pp} to bleed off. To do this, write the Read Memory command prior to issuing the V_{pp_OFF} instruction. Alternatively, the device resets automatically to read mode when V_{pp} drops below $V_{CC} + 2\text{V}$.

Raising V_{pp} to 12V enables the command register. You must wait 100 ns after V_{pp} achieves its steady state value before writing to the command register. Remember that the steady state V_{pp} settling time depends on both the power supply slew rate and the capacitive load on the V_{pp} bus.

3.1.5 OTHER V_{pp} CONSIDERATIONS

The V_{pp} pin is an MOS input which can be damaged by electrostatic discharge (ESD). In OBP applications, an external power source supplies V_{pp} and then is removed. Electrostatic charge can build up on the floating V_{pp} pin. You can solve this problem by one of two means: 1) tie the pin to V_{CC} through a diode and pull-up resistor (Figure 5a) or through a resistor to ground (Figure 5b). With either approach use a 10 K Ω or larger resistor to minimize V_{pp} power consumption.

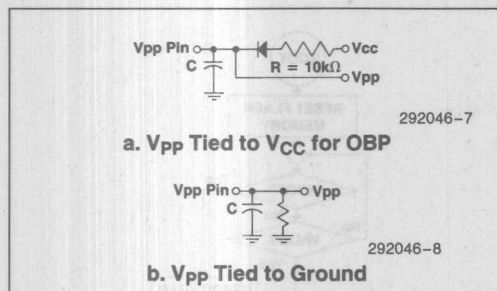


Figure 5

NOTE:

Typically EPROMs require V_{pp} to be within one diode drop of V_{CC} for optimal standby power consumption. Either approach can be used with the flash memory.

ISW applications do not require this ESD protection as most regulators and charge pumps contain a voltage divider on the output stage. A divider provides a resistive path to ground even with the supply turned off. (Note: check the schematics of the V_{pp} supply chosen.) However, if you directly switch the V_{pp} supply, add the resistor to ground; the switch isolates the V_{pp} pin and allows charge to build up.

3.1.6 V_{pp} CIRCUITRY AND TRACE LAYOUT

You should lay out V_{pp} circuitry and traces for high frequency operation since programming power characteristics exhibit an AC current component. Use the following standard power supply design rules:

- Keep leads as short as possible and use a single ground point or ground plane (a ground plane eliminates problems).
- Locate the resistor network (or a regulator) as close as possible to the adjustment pin to minimize noise pick-up in the feedback loop. The resistor divider network should also be as short as possible to minimize line loss.
- Keep all high current loops to a minimum length using copper connections that are as wide as possible. (This will decrease the inductive impedance which otherwise causes noise spikes.)
- Place the voltage regulator as close to the flash memory as practical to avoid an output ground loop. Excessive lead length results in an error voltage across the distributed line resistance.
- Separate the input capacitor return from the regulator load return line. This eliminates an input ground loop, which could result in excessive output ripple.

3.2 Communications—Getting Data to and from the Flash Memory

The flash memory does not care about the origin of the data to be programmed. The data could be downloaded from a serial link, parallel link, disk drive, or generated locally as in data accumulation applications.

While most systems communicate via serial link, sending a font to a printer's flash memory is an example of a parallel interface. In either format, designers must decide whether or not to buffer the incoming data. Error-free serial protocols will require buffering for reconstruction of information packets. With equal capacity of RAM and flash memory in a system, the download time would only be limited by the speed of the communication link.

Both worst case and typical analysis must be done for real time download and un-buffered programming. The maximum transmission rate is 19.2K baud assuming worst case programming times. The time between characters at 19.2K baud is 520 μs; the worst case byte programming time is approximately 0.5 ms (including software overhead). Typical byte programming takes 16 μs which allows for much higher unbuffered transmission rates. However, a single byte can take up to the full 400 μs specified time (plus software overhead), so you should not base transmission rate on typical programming times.

Partial buffering or FIFO schemes can also be implemented to increase transmission rates. An argument for buffering is reduction of interconnect time and costs.

4.0 SOFTWARE DESIGN FOR ISW

Covered in this section are the following software requirements:

- system integration of ISW
- reprogramming considerations for single- and multiple-flash memory based designs.

4.1 System Integration—Boot Code Requirements

Boot code in remote systems should contain various ISW-specific procedures in addition to standard initialization and diagnostic routines.

The most dependable boot code for remote version updates contains some basic communications capability and the ISW reprogramming algorithms. Thus, a data-link disruption while reprogramming would be recoverable. For manufacturing flexibility, this boot memory could be an OBP 256K flash memory.

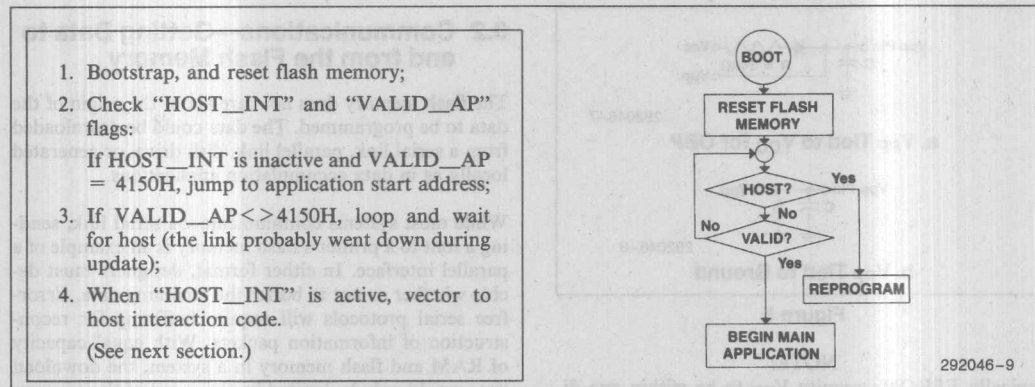


Figure 6. Example of ISW Integration to the Boot Sequence

An alternative to storing these routines in a separate boot device is storing them in the flash memory containing the program code. Prior to erasure, the CPU would transfer the ISW routines to system RAM and execute from there. This type of approach is suitable for battery-operated equipment or systems with back-up power supplies.

The communication link could be disrupted during reprogramming, leaving the device in an unknown configuration. Therefore, the boot code should reset the flash memory and check two ISW flags. The following section discusses the flag check concept.

4.1.1 ISW FLAG CHECK

After resetting the flash memories and initializing other system components, the CPU should check the communications link for a host interrupt. We will call this the HOST_INT flag. Had the communication link gone down prior to completion of downloading, then the host would have to re-establish contact to complete the task.

Assuming no HOST_INT request has been made, the boot protocol then checks a data sequence in the flash memory signifying a valid application (VALID_AP). You program this sequence into the memory array after confirmation of a successful download. If a download is interrupted midway through erasure or programming, then the VALID_AP flag locations will not contain the VALID_AP code. On the next system bootstrap the CPU recognizes this and holds up system boot until valid code is programmed. In Figure 6 an example flag protocol uses the VALID_AP sequence of 4150H (ASCII codes for "AP").

4.2 Communication Protocols and Flash Memory ISW

The remote download communications protocol must guarantee accurate transmission of flash memory in-

structions and program code. This protocol can be as simple as a read-back technique or as complex as an error-free transmission protocol. (See Figure 7 for possible system-level flash memory instructions.)

A simple read-back technique optimizes download for boot code memory needs and ease of implementation. The embedded CPU echoes the flash memory instruction (i.e., Erase or Program) to the host, and waits for a confirmation prior to execution. After programming the update, the remote system checks the update by transmitting it back to the host for confirmation. The remote system then programs the VALID_AP sequence. Note that programming and reading back 64 Kbytes at 19.2K baud takes about 0.57 minutes per direction:

$$(65,536 \text{ bytes}) * (10 \text{ bits/byte}) * (1 \text{ sec}/19.2 \text{ Kbits}) * (1 \text{ min}/60 \text{ sec}) = 0.57 \text{ minutes.}$$

Implementing either software- or hardware-based error-free communications protocol improves transmission efficiency. It eliminates the possibility of errant data being programmed if not buffered and checked, and optimizes the download process for transmission time. Additionally, file compression and decompression routines can improve the transmission rate.

General ISW instructions include:

- STATUS CHECK
- INITIATE REPROGRAMMING
- MOVE ISW ROUTINES FROM FLASH MEMORY TO RAM
(If not resident in separate boot memory)

Data accumulation-specific commands include:

- RETRIEVE DATA
- ERASE FLASH MEMORY

Figure 7. Sample System-Level ISW Instruction Set

The host should request a status update from the remote system prior to sending a reprogramming instruction. Depending on the response, the host may break the link and reconnect later, or it may send an erasure or data-upload command. This type of handshaking is necessary when system downtime for reprogramming might not be acceptable. An example of this is an automated factory where robots handle caustic chemicals.

4.3 Data Accumulation Software Techniques

Data can be accumulated in a remote environment with flash memory and then uploaded to a host computer for manipulation. You can adapt various standard data-logging techniques for use with flash memory. With any technique, you determine the next available memory location by reading for erased data (0FFH). This address would only be located once on system bootstrap and then recalled from RAM and incremented as needed.

Given a repeating data string of known length and composition, program start and stop codes at either end of the string. Do not pick 00H or 0FFH data for these codes because they are used during erasure. The start and stop codes enable the CPU to differentiate between available memory for logging and logged data equal to 00H or 0FFH.

For non-regular data input, you can address this same issue by programming the logged data followed by the variable identifier. Again, do not pick 00H or 0FFH data for the variable identifiers.

With any technique, the host computer separates and manipulates the data after the uploading operation.

4.4 Reprogramming Routines

Intel's ETOX flash memories provide a cost-effective updatable, non-volatile code storage medium. The reliability and operation of the device is based on the use of specified erasure and programming algorithms.

Intel offers reprogramming software drivers to make it easy for you to design and implement flash memory applications. The software is designed around the CPU-family architectures and requires minimal modification to define your system parameters. For example, you supply the memory width (8-bit, 16-bit, or 32-bit), system timing, and a subroutine for control of V_{pp} .

If you prefer to implement the algorithms yourself, they are outlined in the device data sheets. Command register instructions required for the various operations are included in the data sheet flow charts.

The following sections describe both single-device and multiple-device parallel reprogramming implementations.

4.4.1 Quick-Erase Algorithm

Flash memories chip-erase all bits in the array in parallel. The erase time depends on the V_{pp} voltage level (11.4V–12.6V), temperature, and number of erase/write cycles on the part. See the device data sheets for specific parametric influences on reprogramming times.

Note that prior to erasing a flash memory device the processor must program all locations to 00H. This equalizes the charge on all memory cells insuring uniform and reliable erasure.

Algorithm Timing Delays

The Quick-Erase algorithm has three different time delays:

- 1) The first is an assumed delay when V_{pp} first turns on. The capacitors on the V_{pp} bus cause an RC ramp. After switching on V_{pp} , the delay required is proportional to the number of flash memory devices times $0.1 \mu\text{F}/\text{device}$. V_{pp} must reach its final value 100 ns before the CPU writes to the command register. Systems that hardwire V_{pp} to the device can eliminate this delay.
- 2) The second delay is the "Time Out TEW" function, where TEW is the erase timing width. The function occurs after writing the erase command (the second time) and before writing the erase-verify command. The erase-verify command or the integrated stop timer internally stops erasure.
TEW for ETOX II flash memories is a minimum of 10 ms. This delay can be either software or hardware controlled. Either way, the minimum nature of the timing specification allows for interrupt-driven timeout routines. Should the interrupt latency be longer than the minimum delay specification, the stop timer halts erasure.
- 3) The third delay in the erase algorithm is a $6 \mu\text{s}$ time out between writing the erase verify command and reading for 0FFH. During this delay, the internal voltages of the memory array are changing from the

erase levels to the verify levels. A read attempt prior to waiting 6 μ s will give false data—it will appear that the chip does not erase. Repeatedly trying to erase verify the device without waiting 6 μ s will cause over-erasure. This delay is short enough that it is best handled with software timing. Again, note that the delay specification is a minimum.

High Performance Parallel Device Erasure

In applications containing more than one flash memory, you can erase each device serially or you can reduce total erase time by implementing a parallel erase algorithm.⁷ You save time by erasing all devices at the same time. However, since flash memories may erase at different rates, you must verify each device separately. This can be done in a word-wise fashion with the command register Reset command and a special masking algorithm.

Take for example the case of two-device (parallel) erasure. The CPU first writes the data word erase command 2020h twice in succession. This starts erasure. After 10 ms, the CPU writes the data word verify command A0A0h to stop erasure and setup erase verifica-

tion. If both bytes are erased at the given address, then the CPU increments the address (by 2) and then writes the verify command A0A0h again. If neither byte is erased, then the CPU issues the erase sequence again without incrementing the address.

Suppose at the given address only the low byte verifies FFh data? Could the whole chip be erased? The answer is yes. Rather than check the rest of the low byte addresses independently of the high byte, simply use the reset command to mask the low byte from erasure and erase verification on the next erase loop. In this example the erase command would be 20FFh and the verify command would be A0FFh. Once the high byte verifies at that address, the CPU modifies the command back to the default 2020h and A0A0h, increments the address by 2, and writes the verify command to the next address.

See Figure 8 for a conceptual view of the parallel erase flow chart and Appendix D for the detailed version. These flow charts are for 16-bit systems and can be expanded for 32-bit designs.

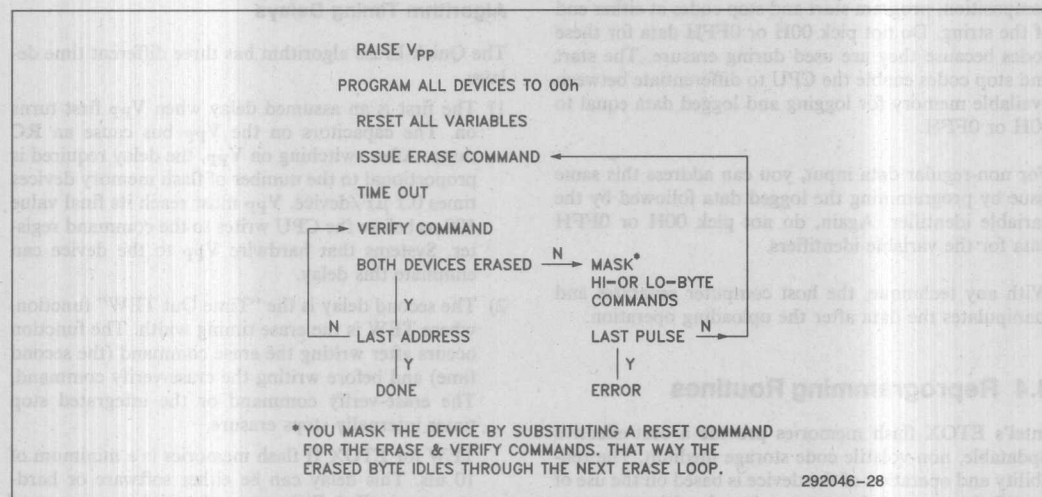


Figure 8. High Performance Parallel Erasure (Conceptual Overview)

7. Parallel Erasure and Programming require appropriate choice of V_{pp} supply to support the increased power consumption.

4.4.2 Quick-Pulse Programming Algorithm

Flash memories program with a modified version of the Quick-Pulse Programming algorithm used for U.V. EPROMs. It is an optimized closed-loop flow consisting of 10 μ s program pulses followed by byte verification. Most bytes verify after the first pulse, although some may require more passes through the pulse/verify loop. As with U.V. EPROMs, this algorithm guarantees a minimum of ten years data retention. See the device data sheets for more details on the programming algorithm.

Algorithm Timing Delays

The Quick-Pulse Programming algorithm has three different time delays:

- The first and third—V_{pp} set-up and verify set-up delays—are the same as discussed in the erasure section. In this case the third delay is for the transition between writing the Program Verify command and reading for valid data.

- The second delay is the “Time Out 10 μ s” function, which occurs after writing the data and before writing the program-verify command. This write command internally stops programming. The section entitled “Pulse Width Timing Techniques” gives 86-family assembly code for generating a 10 μ s timer routine.

High Performance Parallel Device Programming

Software for word- or double-word programming can be written in two different manners. The first method offers simplicity of design and minimizes software overhead by using a byte programming routine on each device independently. Here you increment the address by 2 or 4 when addressing 1 of 2 or 4 devices, respectively. The second method offers higher performance by programming the word or double-word data in parallel. This method manipulates the command register instructions for independent byte control. See Figure 9 for conceptual 2-device parallel programming flow chart and Appendix E for the detailed version.

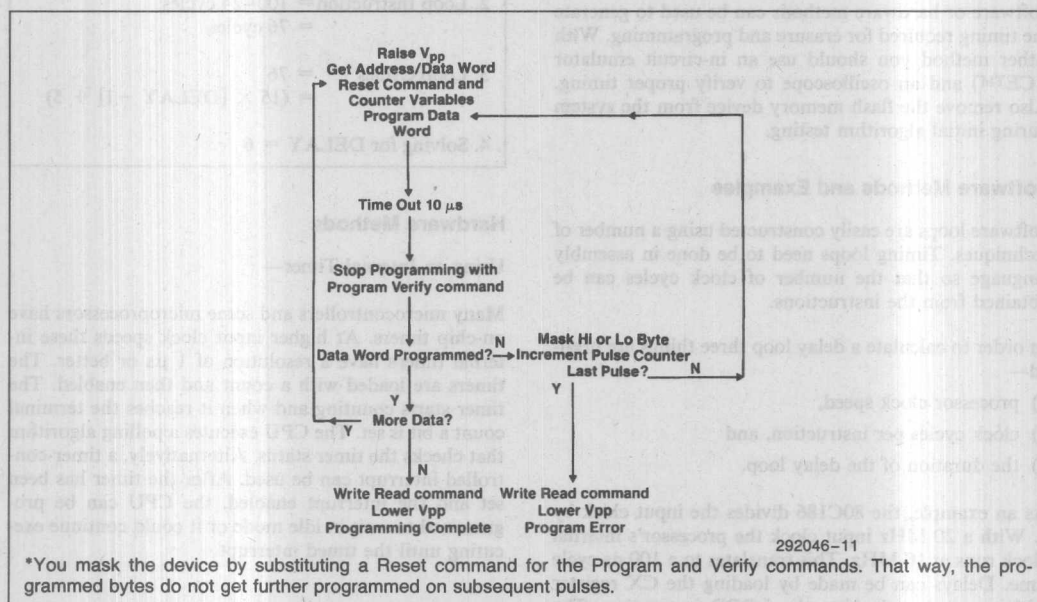


Figure 9. Parallel Programming Flow Chart (Conceptual Overview)

NOTE:

Word or double-word programming assumes 2 or 4 8-bit flash memory devices.

Parallel Programming Algorithm Summary:

- Decreases programming time by programming 2 flash memories (16 bits) in parallel. The algorithm can be expanded for 32-bit systems.
- Eliminates tracking of high/low byte addresses and respective number of program pulses by directing the CPU to write data-words (16-bit) to the command register.
- Maintains word write and word read operations. Should a byte on one device program prior to a byte on the other, the CPU continues to write word-commands to both devices. However, it deselects the verified byte with software commands. An alternative is to independently program high and low bytes using hardware select capability.

4.4.3 Pulse Width Timing Techniques

Software or hardware methods can be used to generate the timing required for erasure and programming. With either method you should use an in-circuit emulator (ICE™) and an oscilloscope to verify proper timing. Also remove the flash memory device from the system during initial algorithm testing.

Software Methods and Examples

Software loops are easily constructed using a number of techniques. Timing loops need to be done in assembly language so that the number of clock cycles can be obtained from the instructions.

In order to calculate a delay loop three things are needed—

- 1) processor clock speed,
- 2) clock cycles per instruction, and
- 3) the duration of the delay loop.

As an example, the 80C186 divides the input clock by 2. With a 20 MHz input clock the processor's internal clock runs at 10 MHz. This translates to a 100 ns cycle time. Delays can be made by loading the CX register with a count and using the LOOP instruction. The

LOOP instruction takes 16 clock cycles to execute per pass. It decrements the CX register on each pass and jumps to the specified operand until CX equals zero.

When writing a delay loop consider all instructions between the start and end of the delay. If a macro is written that delays 10 μ s, add the clock cycles for all instructions in the macro.

Here is an example of a 10 μ s delay and the calculation of the constant required for a 10 MHz 80C186.

WAIT_10 μ s:

```

push cx           ;10 clock cycles
mov cx,DELAY      ;4 clock cycles
loop $            ;see calculation
pop cx            ;10 clock cycles

```

1. Start to End = 10 μ s/cycle time
 = 10 μ s/100 ns
 = 100 cycles
2. Loop Instruction = 100–24 cycles
 = 76 cycles
3. Loop Cycles = 76
 = (15 \times [DELAY – 1] + 5)
4. Solving for DELAY = 6

Hardware Methods**Using an Internal Timer—**

Many microcontrollers and some microprocessors have on-chip timers. At higher input clock speeds these internal timers have a resolution of 1 μ s or better. The timers are loaded with a count and then enabled. The timer starts counting and when it reaches the terminal count a bit is set. The CPU executes a polling algorithm that checks the timer status. Alternatively, a timer-controlled interrupt can be used. After the timer has been set and the interrupt enabled, the CPU can be programmed to wait in idle mode or it could continue executing until the timed interrupt.

interrupt request (interrupt latency). This is important when figuring the timer value, because the time seen by the part will be the programmed delay plus the minimum interrupt latency time.

The 80C186 has three 16-bit timers on-chip. Timer #2 can be a prescaler for the other two timers, which extends timers #0 and #1 range out to 2^{32} . By using two timers, 10 μ s pulses and 10 ms pulses can be easily achieved.

Using an External Timer—

External timers can take many forms. One popular example is the 82C54 (CHMOS Programmable Interval Timer) which has three 16-bit timers on-chip. One timer can be used as a prescaler for the others so that a count of 2^{32} can be achieved as with the 80C186 internal timers.

5.0 SYSTEM DESIGN EXAMPLE: AN 80C186 DESIGN

A general purpose controller and/or data acquisition system was built to demonstrate 86-based ISW. The 80C186 CPU drives the system, which contains 16 Kbytes of EPROM (two 27C64's), 64 Kbytes of flash memory (two 28F256A's), 64 Kbytes of SRAM (two 32K x 8's) three 8-bit ports (82C55A), one serial port (82510), and a 5V to 12.0V DC/DC converter. Three 74HC573's demultiplex the address/data bus and latch the byte high enable line (BHE#) and the status lines (if needed). Two data transceivers (74HC245) simulate the worst case data path for a system requiring added drive capability. If the transceivers are not needed they can be replaced with wired headers. See Appendix F for detailed schematics parts list, and changes for the 28F512 or 28F010.

The 80C186 reset (output) drives the reset input on the 82510, 82C55A, and the OE# inputs on the address latches and data transceivers. The reset line goes inactive 5 clock cycles before the first code fetch. Also, the CPU's write signal is split into byte-write-high and byte-write-low to allow for byte or word writes.

The 80C186 has on-chip memory and peripheral chip selects. Two of the memory chip selects are dedicated. One is the Upper Chip Select (UCS#, dedicated for the boot area) and the second is the Lower Chip Select (LCS#, for the interrupt vector table area). See the memory map in Figure 10.

	Boot	flash memory algo's, etc.
UCS		FC000H
	Application	Version update code, Data Accumulation storage, etc.
MCSO		40000H
	RAM	Vector table, Stack, Buffers, etc.
LCS		0000

Figure 10. 80C186 Memory Map

The permanent code was placed in an EPROM in the UCS memory segment; this code includes routines for hardware initialization, communications, data uploading and downloading, erasure and programming algorithms, I/O drivers, ASCII to binary conversion tables, etc. This would be useful for systems reconfigured for different communication protocols as the last step prior to shipment.

Code and constants that might change are placed in the 64 Kbytes of flash memory. Application examples include operating systems, code for rapidly advancing biomedical technologies such as blood test software, engine-control code and parameters, character fonts for printers, postage rates, etc. The RAM is used for the interrupt table, stack, variable data storage, and buffers.

The three 8-bit ports on the 82C55A peripheral controller can be used for control and/or data acquisition. It powers-up with all port pins high. Similarly, all port pins go high after warm resets as well. Because the pins are high after a power-up/reset, an open collector inverter was used to control the MOSPOWER switch which in turn controls V_{pp} . You must drive the FET switch to one rail or the other to guarantee its low on-resistance. V_{pp} is turned off during power-up or reset as a hardware write protection solution. The DC/DC converter supplies V_{pp} .

The 82510 is a flexible single channel CHMOS UART offering high integration. The device off-loads the system and CPU of many tasks associated with asynchronous serial communications.

The part can be used as a basic serial port for the host serial link, or can be configured to support high speed modem applications. For more information on the 82510 see the 82510 data sheet and AP-401 "Designing with the 82510 Asynchronous Serial Controller".

Software was written to download code and data parameters (code updates) from a PC to the demo board through the PC's COM1 port (serial port). The system also can upload data (remote data acquisition) to the PC via the same link.

Once the download code and data has been programmed it can not be lost, even if power should fail. This is because Intel's ETOX II flash memory technology is based on EPROM technology and does not need power to retain data.

The end result: rugged, solid state, low power nonvolatile storage.

6.0 SUMMARY

Intel's flash memories offer designers cost-effective alternatives for remote version updates or for reliable data accumulation in the field or factory. Designers will also benefit from time savings in any kind of code development—no 15 minute waits for U.V. EPROM erasure.

This application note covers the basics of in-system writing to flash memories and can be used as a check list for systems other than the 80C186 design shown. The basic concepts remain the same: a CPU controls the reprogramming operations; a 12V supply must be applied to the flash memory for erasure and programming; and a communications link connects the host to the remote system and supplies the code to be programmed.

6.0 SYSTEM DESIGN EXAMPLE: AN 80C186 DESIGN

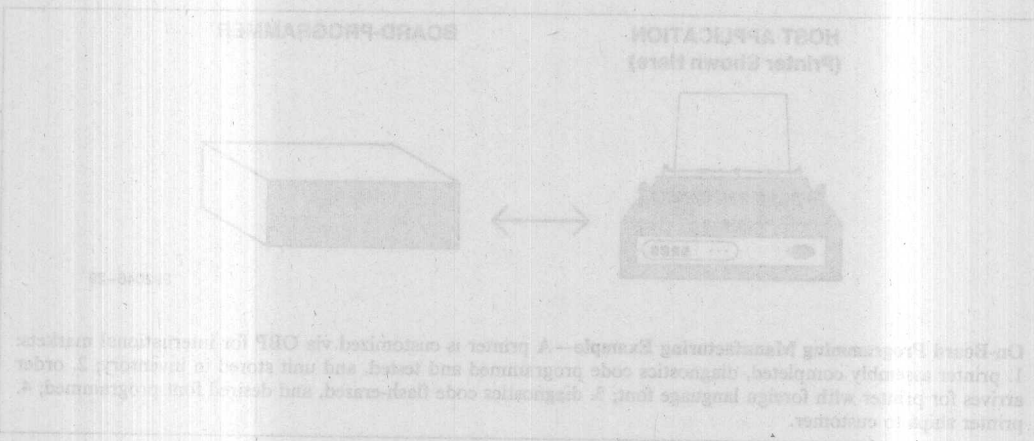
A general purpose controller and/or data acquisition system was built to demonstrate 8K-based 12V. The 80C186 CPU drives the system which contains 16 Kbytes of EPROM (two 8Kx16), 64 Kbytes of flash memory (two 32Kx16), 64 Kbytes of SRAM (two 32Kx8), 8-bit ports (80C23A), and serial port (82510) and a 5V to 12V DC/DC converter. Three 80C173 decouplers are connected to the 80C186 and the 80C23A and the 80C173 decouplers the address/data bus (80C186) and the 80C23A. Two data transceivers (80C245) are used to connect the 80C186 to the 80C23A and the 80C23A to the 80C173. The 80C173 decouplers are not needed if they can be replaced with word buffers. See Appendix B for detailed schematic data and changes for the 80C186 or 80C188.

The 80C186 reset (pin 1) drives the reset input on the 82510, 80C23A, and the 80C173. The reset line goes through a 10K resistor to the 80C186. The 80C173 decouplers are also connected to the 80C186. The 80C173 decouplers are not needed if they can be replaced with word buffers. See Appendix B for detailed schematic data and changes for the 80C186 or 80C188.

The 80C186 has on-chip memory and peripheral chips. Two of the memory chips are dedicated. One is the Upper Chip Select (UCS*) dedicated for the port (pin 1) and the other is the Lower Chip Select (LCS*) for the interrupt vector table (pin 10). memory map is shown in Figure 10.

APPENDIX A ON-BOARD PROGRAMMING DESIGN CONSIDERATIONS

1. With on-board programming, non-volatile memory is programmed while socketed or soldered on the location board. After the device has been programmed, the programming method is also called in module or in circuit programming. This method is described by some major corporations since 1981. See sidebar on following pages for more information on I.V. EPROM usage.



5

INTEL'S FLASH MEMORY—DESIGNED TO MEET YOUR OBP NEEDS

Intel's flash memory simplifies OBP code updates by offering designers the command register architecture. As described in section 2.5, the architecture offers the full reliability of EPROM on-board programming without the hassle of blowing V_{CC}.

Unlike EPROM OBP, flash memory enables V_{CC} to remain at 5.0V throughout all operations. Intel's circuitry derives the erase and program voltages from the voltage on V_{CC} rather than from V_{PP}. These verify modes enable use of a single V_{CC} for the entire board, as opposed to the two buses needed for U.V. EPROM OBP (see sidebar entitled EPROM OBP).

5 Volt V_{CC} Erase and Programming Verification

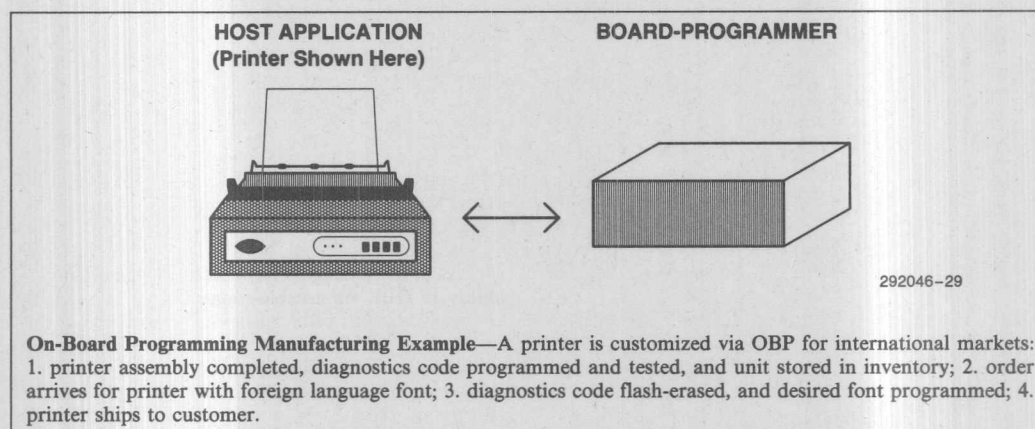
INTRODUCTION

On-board programming¹ (OBP) with Intel's flash memory provides designers with cost reduction capabilities for alterable code storage designs. When used in conjunction with on-board programming, flash memory presents opportunities for savings in two areas: greater testability in the factory, which translates to improved outgoing quality and reduced return rate; and quicker, more reliable field updates, which translates to decreased product support cost.

1. With on-board programming, non-volatile memory is programmed while socketed or soldered on the application board, rather than before hand as a discrete component. This programming method is also called in-module or in-circuit programming, and has been practiced by some major corporations since 1981. See sidebar on following pages for more information on U.V. EPROM OBP usage.

This appendix:

- outlines the design considerations associated with on-board programming, and the improvements afforded by Intel's flash memory;
- offers guidelines for converting current 64K EPROM OBP designs;
- designs an 8-bit system for on-board programming;
- suggests some 16-bit flash design considerations; and offers information on OBP equipment and vendors.



INTEL'S FLASH MEMORY—DESIGNED TO MEET YOUR OBP NEEDS

Intel's flash memory simplifies OBP code updates by offering designers the command register architecture. As described in section 2.2, this architecture offers the full reliability of EPROM off-board programming without the hassles of elevating V_{CC} .

5 Volt V_{CC} Erasure and Programming Verification

Unlike EPROM OBP, flash memory enables V_{CC} to remain at 5.0V throughout all operations. Internal circuitry derives the erasure and programming verification levels from the voltage on V_{PP} rather than from V_{CC} . These verify modes enable use of a single V_{CC} bus for the entire board, as opposed to the two buses needed for U.V. EPROM OBP. (See sidebar entitled EPROM OBP).

EPROM OBP

EPROM OBP has been a proven manufacturing technique since 1981. Ingenuity and clever circuit design have enabled manufacturers to overcome the hurdles associated with OBP and enjoy the benefits.

In many cases, Intel's flash memory simplifies today's solutions and offers new capabilities to advance the state of OBP technology. The following paragraphs outline the hurdles and a few of the solutions in use today.

EPROMs require program verification at an elevated V_{CC} to insure long-term data retention. PROM programmers easily accommodate this requirement, and it is generally invisible to the end-user.

REPLACING CURRENT EPROM OBP DESIGNS WITH FLASH MEMORY

Hardware Considerations

A slight hardware modification is required to adapt most of the current EPROM OBP designs for use with Intel's flash memory. Simply convert the EPROM memory sites from 28 to 32 pins. All other board-design criteria used for EPROM OBP apply to flash memory as well. (For discussions of these criteria see section entitled New OBP Designs).

Standard EPROM OBP requires the board designer to bus PGM# to the edge connector. With flash memories' command register architecture, this same trace enables electrical erasure and programming, only now the line is called Write Enable (WE#). The timing for WE# is similar to that of read accesses, although that is handled via software changes.

Another potential hardware change is on the board programmer side of the design—the V_{pp} supply. Many EPROMs program with 12.5-13.0V V_{pp} supplies. Intel's ETOX II flash memory requires 11.4-12.6V V_{pp}. This change should not be an issue since the V_{pp} supply on many board programmers is programmable.

Mixed memory systems containing both conventional U.V. EPROM and flash memories require special consideration. This type of memory design requires separation of the Chip Enable (CE#) control lines between the EPROM and flash devices to allow for independent reprogramming control and access. The PGM# and

WE# lines can be common if the board programmer can give the appropriate timings to either type of device.

Software Considerations

Manufacturers who program EPROMs on-board today will need new board-programmer software to take advantage of flash memory's feature set, specifically software for the Quick-Erase and Quick-Pulse Programming algorithms.

Benefits of Converting Your EPROM OBP Design to Flash

The most pressing reason to convert from a standard EPROM to flash memory is the total cost savings. To appreciate this, you must consider your way of doing business at the board and system levels—from the factory to installation and repair in the field. In the factory, boards can be tested with a diagnostics program in the flash memory and then erased and reconfigured for shipment in the same step. Improved testing will decrease the probability of field failures and costly customer returns. Simplified test and rework methods will decrease your inventory holding costs. Also, if in the process of converting to flash memory you include the ability to OBP via a cable-connector, service calls for code updates will be quicker, more reliable, and cost less money. Your serviceman would simply connect the programming equipment to the system without dismantling it to remove the EPROMs. (See section entitled The System/Board-Programmer H/W Connection for details.)

5

EPROM OBP (cont'd)

With OBP, the EPROM board-programmer handles the elevated-V_{CC} requirement easily as well. However, when V_{CC} is greater than 5V, logic devices populating the same board may draw excessive current and not operate predictably.

One solution to this issue involves running separate V_{CC} traces to the board's edge connector—one for EPROM programming, and one for powering up the rest of the board.

A second consideration when designing for EPROM OBP has been accessing manufacturer and device codes.

The identifier mode requires forcing A9 to 12V. This translates to adding extra isolation, which implies the increased costs of buffers and extra board space.

NEW OBP DESIGNS

Design Considerations

As with EPROM in-circuit programming, flash memory board programming requires the use of a board-programmer. Unlike U.V. erasure for standard EPROM OBP, electrical erasure enables flash memory OBP without removing the board from the system.

We will look at designing a board that is to remain powered-up in the system during erasure and reprogramming. The key concept is to design the board in such a way that the programmer can take control of the system during code updates. The implementation of such a design is straightforward, easy, and suited to automated production assembly.

Taking Control

The board-programmer needs to take control of the system's address bus, data bus, control lines, etc. to update the code without damaging the system. (See Figure 2. System to Board-Programmer Interface.) Taking control simply means isolating the rest of the system from these lines.

Various methods of isolating the memory from the system include using tristate buffers, latches, or even the capabilities designed into microprocessors (μP) and microcontrollers (μC). For example, Intel's 86-based μP family has HLD/HLDA signals that were set-up for multiprocessor system designs where bus control is a major concern. The HLD signal, when acknowledged, tristates the address, data, and control lines. Although not designed for multiprocessor environments, Intel's MCS®-51 and MCS-96 microcontroller families have Reset capabilities to help simplify this same task.

One issue to be aware of when using a CPU's reset control function is that it may switch from the reset to active condition at a non-standard logic level. This only presents a problem if the address/data buffer takes longer to activate than the CPU, and the CPU attempts to fetch code from a memory device isolated from it.

One approach to insure successful programming takeover (i.e. without bus contention) is to have the board-programmer's lines in a high impedance state during connection to the system. Once connection to the system has been secured, the serviceman could hit a button on the board-programmer to start the system takeover procedure. Then when total control has been established, the programmer would commence with erasure and reprogramming.

Aside from the flash device's isolation from the system, various CPU control lines (MEMRD#, WE#, PSEN#, etc.) may need isolation as well. If active during Reset, these lines may put the CPU into an unspecified state. When designing a board for OBP, check the $\mu C/\mu P$ data sheets carefully for any special reset conditions.

Printed Circuit Board Guidelines for V_{CC} and V_{PP}

Programming conventional EPROM and flash memories takes 30 mA of current on V_{CC} and V_{PP} , due to the nature of hot-electron injection. Most of the charge transfers to the memory cell's floating gate in a short current spike during the first pulse. You should design both the V_{CC} and V_{PP} traces with A.C. current spikes in mind. Wherever possible, limit the inductance by widening the two traces. Bypass capacitors (0.1 μF) should be placed as close as possible to the memory device's V_{CC} and GND pins, as well as the devices V_{PP} and GND pins. The capacitor on V_{CC} decreases the power supply droop. The capacitor on V_{PP} supplies added charge, and filters and protects the memory from high frequency over-voltage spikes².

2. For a complete discussion of electrical noise, grounds, power supply distribution and decoupling see Ap-74—High Speed Memory System Design Using the 2147H, and AP-125—Designing Microcontroller Systems for Electrically Noisy Environments.

EPROM OBP (cont'd)

Some users of OBP get around this issue by programming all EPROMs with a common algorithm. However, this practice compromises the device's reliability, and should not be done.

A better solution than ignoring the identifier is to choose a qualified EPROM vendor and program with its algorithm only.

One subtle concern with EPROM OBP that designers often overlook is U.V. board erasure.

→ U.V. EPROM board erasure requires removal of the board from its host system. This incurs the hidden costs of labor, lower yields due to handling, and the reliability risks of dismantling a system. Flash memory decreases these costs by enabling a greater degree of factory automation, and increases the flexibility afforded by OBP.

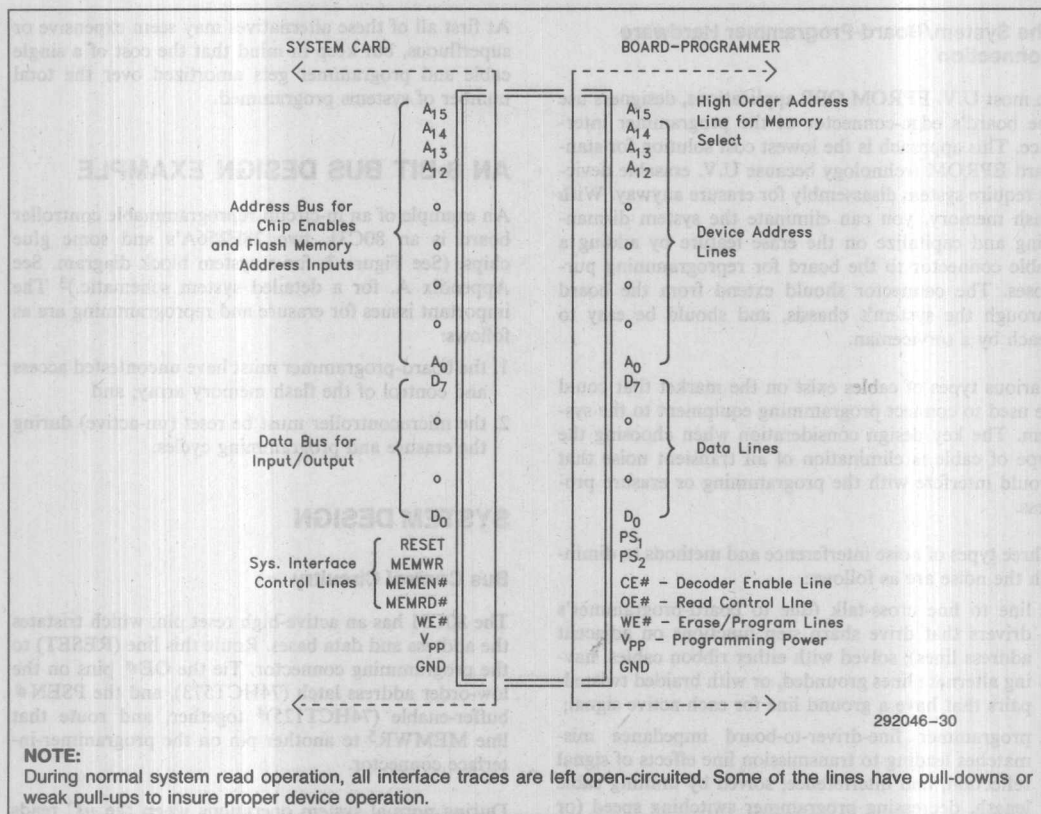


Figure 2. System to Board-Programmer Interface

EPROM OBP (cont'd)

→ Special U.V. board erasers must be purchased, at significant costs and with limited throughput. A low-end U.V. bulb costs \$75-\$100 each. A U.V. board eraser system could cost upwards of \$10,000, with recurring costs of light bulbs and energy. Thus, the cost of U.V. erasure is often under-estimated.

→ Although portable board programmers are commercially available, U.V. lights by nature are not very rugged, and are not suited for out-of-factory code updates. This complicates field service.

→ Erasure can be easily controlled in a lab environment; however, it is not as clear on the manufacturing floor which label to remove for U.V. erasure, because parts other than EPROMs have windows (i.e. EPLD's, micro-controllers with embedded EPROM memory, etc.)

The System/Board-Programmer Hardware Connection

In most U.V. EPROM OBP applications, designers use the board's edge-connector as the programmer interface. This approach is the lowest cost solution for standard EPROM technology because U.V. erasable devices require system disassembly for erasure anyway. With flash memory, you can eliminate the system dismantling and capitalize on the erase feature by adding a cable connector to the board for reprogramming purposes. The connector should extend from the board through the system's chassis, and should be easy to reach by a serviceman.

Various types of cables exist on the market that could be used to connect programming equipment to the system. The key design consideration when choosing the type of cable is elimination of all transient noise that would interfere with the programming or erasure process.

Three types of noise interference and methods to diminish the noise are as follows:

1. line to line cross-talk (due to board-programmer's drivers that drive sharp step functions on adjacent address lines); solved with either ribbon cables, having alternate lines grounded, or with braided twisted-pairs that have a ground line for each active signal;
2. programmer line-driver-to-board impedance mismatches leading to transmission line effects of signal reflection, and interference; solved by limiting cable length, decreasing programmer switching speed (or allowing longer settling time between address switches) or by using matched line drivers on the programmer and high impedance buffers on the board end, or by using series termination resistors on the driving end of the cable (i.e.—board-programmer end, with the exception of the bi-directional data bus which needs series resistors at both ends);
3. rf pick-up in electrically noisy environments; use either shielded cable such as coax, ribbon cable with solid copper ground plane, or a new type that has recently become available called Flex cable.

Braided twisted-pair cables when kept under three feet in length generally reduce cross-talk to acceptable levels. This type of cable offers the most cost-effective solution which works well in most applications. Depending on the environment, the programmer and your design, you may need a combination of solutions, such as braided twisted-pairs with series termination.

3. Note that the flow-through latch on the data bus is not needed with the 80C31, but is drawn as an example for CPU's that can not tristate their data bus.

4. The isolation buffer is required on PSEN# in this design because the 80C31 goes into unspecified states when the Reset and PSEN# lines are active simultaneously. To avoid any possible problems, buffer PSEN#.

5. MEMWR = > bus isolation control of PSEN# and the data bus.

At first all of these alternatives may seem expensive or superfluous, but keep in mind that the cost of a single cable and programmer gets amortized over the total number of systems programmed.

AN 8-BIT BUS DESIGN EXAMPLE

An example of an in-circuit reprogrammable controller board is an 80C31, two 28F256A's and some glue chips. (See Figure 3. for a system block diagram. See Appendix A. for a detailed system schematic.)³ The important issues for erasure and reprogramming are as follows:

1. the board-programmer must have uncontested access and control of the flash memory array; and
2. the microcontroller must be reset (un-active) during the erasure and programming cycles.

SYSTEM DESIGN

Bus Control Circuitry

The 80C31 has an active-high reset pin, which tristates the address and data buses. Route this line (RESET) to the programming connector. Tie the OE# pins on the low-order address latch (74HCT573), and the PSEN# buffer-enable (74HCT125)⁴ together, and route that line MEMWR⁵ to another pin on the programmer-interface connector.

During normal system operations when the μ C reads program code from the 28F256 devices, the pull-down on MEMWR keeps the address latches and PSEN# buffer active. During flash memory OBP, the board-programmer drives MEMWR active-high, which disables these outputs, and isolates the address bus and PSEN# from the programming signals.

The board-programmer must independently control the RESET and MEMWR traces because they disable at different V_{IL} values (2.5V for RESET vs 0.8V for MEMWR). If controlled by the same 5V supply, on power-up or after a reset condition the μ C would try to execute code while still isolated from its code source—specifically before the address latches and PSEN# buffer activate.

Address Decode Circuitry

This design shows two 28F256A flash memories. Systems with more than one memory device typically decode the CPU's high-order address to select a particular device.



Figure 3. System Block Diagram

This is accomplished as illustrated. When A15 is low, the lower 32K bytes are selected. The output of the inverter drives the other 28F256A's chip enable. This type of memory architecture promotes power savings by disabling all memories but the one being addressed.

To accomplish this two-line memory control architecture, route the inverter's input A15 to the 80C31 and to the programmer interface connector.⁸ The board-programmer controls the inverter's output enable with MEMEN#.⁹ The MEMEN# line performs the function normally performed by CE# in component programming. When driven to a logic "1" level MEMEN# pulls the inverter's output high. This deselects all memory devices controlled by that I.C. During normal read and standby operations, the pull-down on MEMEN# keeps the decoder enabled.

Erasure and Programming Control Circuitry

In this design, V_{pp} and WE# are active only during reprogramming. At other times, the two inputs would be inactive. Simply tie the WE# line to V_{CC} through a pull-up resistor. The pull-up limits the current to the board programmer during reprogramming. (Recall that WE# is active low.) Flash memories allow V_{pp} to be at 12V, V_{CC} or ground for read operations. This design ties V_{pp} to V_{CC} through a diode and resistor to allow for EPROM OBP compatibility. If this option is not required, simply tie V_{pp} to ground through a current-limiting pull-down resistor.

Returning Control to the Host System

The board-programmer should return system-control to the host processor in an organized manner. First it should lower V_{pp} from 12V to 5V, or ground. Then the board programmer should place its address and data

buses into a high impedance state. Next PS2, which controls MEMWR should be tristated thus disabling the PSEN#/Address latch isolation. Finally the board-programmer should switch PS1, which drives the RESET line to reactivate the μ C. This sequence guarantees that the μ C will begin operation at a known program code location.

16-BIT BUS DESIGN CONSIDERATIONS

An example of an On-Board programmable 16-bit system board would be an 80C186 microprocessor, two 28F010 flash memories, RAM, and some glue chips. The basic hardware design considerations would be the same as those in the previously discussed 8-bit bus example.

There are a few issues with 16-bit designs that do not arise in 8-bit designs. For the programmer to take control of the system, it must tristate and reset the μ P as well as tristate the bus buffers and latches. The HOLD and RESET lines of Intel's 86-based family of microprocessors have been designed with bus isolation in mind for use in multiprocessor systems.

The designer has two options for erasing and programming the high and low bytes of the flash memory array independently.

- 1) The designer can route two WE# lines to the programmer connector—BYTE HIGH WE# and BYTE LOW WE#.
- 2) The reprogramming software can follow the masking procedure shown in section 4.4. This method allows a common WE# line for the high and low bytes.

8. Note the lack of isolation buffers between the 80C31's high order addresses (Port 2) and the board-programmer interface, compared to the latch separating the low order addresses (Port 0) and the interface. In this design example, we make use of the 80C31's ability to tristate these ports, so no isolation is needed for any of the addresses. The latch on Port 0 is for the time-multiplexed address/data architecture of this microcontroller, and not specifically for isolation.

9. MEMEN = memory enable, active low.

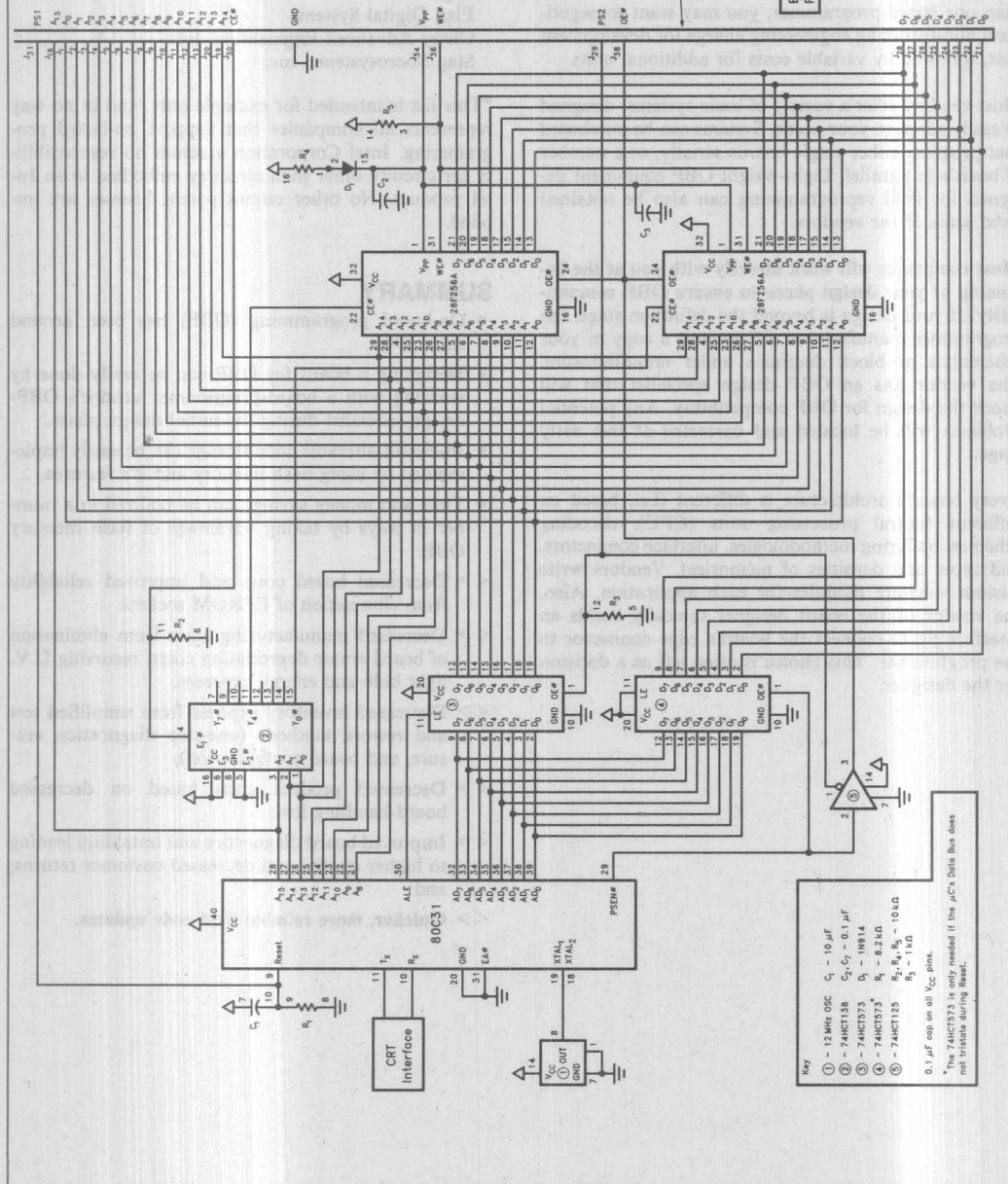


Figure 4. Detailed 8-Bit Bus Design Schematic

OBP EQUIPMENT AND VENDORS

If you are considering OBP for your next design, and have not used on-board programming before, you will need to choose a board-programmer vendor. Various suppliers offer OBP systems; therefore, it is well worth it to send out requests for programming support bids. If your production volume justifies the purchase of more than one board-programmer, you may want to negotiate a non-recurring engineering charge for development cost, followed by variable costs for additional units.

Most vendors offer a variety of basic systems, designed to easily adapt to your needs. Systems can be purchased that program either single boards serially, or a number of boards in parallel. Light-weight OBP equipment designed for field reprogramming can also be obtained from some of the vendors.

Most companies will work directly with you at the beginning of your design phase to ensure OBP compatibility. If your design is beyond the definition stage, the programmer manufacturer will request a copy of your schematics or block diagrams under non-disclosure. The vendor has an OBP design specialist that will check the design for OBP compatibility. Any potential problems will be located and corrected at this early stage.

Every board's architecture is different (i.e., based on different central processing units (CPU), decoding schemes, buffering methodologies, interface connectors, and types and densities of memories). Vendors write custom software modules for each application. Also, the vendor or the board designer typically builds an interface jig to connect the board's edge connector to the programmer. This choice is often left as a decision for the designer.

Partial List* of Companies Selling Board-Programmers

Following are a few of the companies who offer on-board programming solutions today:

Data I/O Corp.
Digelec
Elan Digital Systems
Oliver Advanced Engineering, Inc.
Stag Microsystems, Inc.

*This list is intended for example only, and in no way represents all companies that support on-board programming. Intel Corporation assumes no responsibility for circuitry other than circuitry embodied in an Intel product. No other circuit patent licenses are implied.

SUMMARY

- On-board programming (OBP) has been around since 1981.
- Designing a board for OBP can be easily done by working with a board-programmer vendor's OBP-design-specialist during the initial design phase.
- In-circuit alterable code storage can be easily implemented by using flash memory and its features.
- Time and money savings can be realized in a number of ways by taking advantage of flash memory OBP:
 - <> Decreased board costs and improved reliability from elimination of EPROM sockets;
 - <> Decreased manufacturing costs from elimination of board eraser depreciation costs, recurring U.V. light bulb and energy expenses;
 - <> Decreased inventory expense from simplified test and rework methods (one-step diagnostics, erasure, and board configuration);
 - <> Decreased product costs based on decreased board-handling loss;
 - <> Improved board diagnostics and testability leading to higher quality and decreased customer returns; and
 - <> Quicker, more reliable field code updates.

APPENDIX B V_{PP} GENERATION CIRCUITS

Circuit #1—Regulation from a higher voltage

Circuit #2—Regulation from a higher voltage

Circuit #3—Regulation from a higher voltage

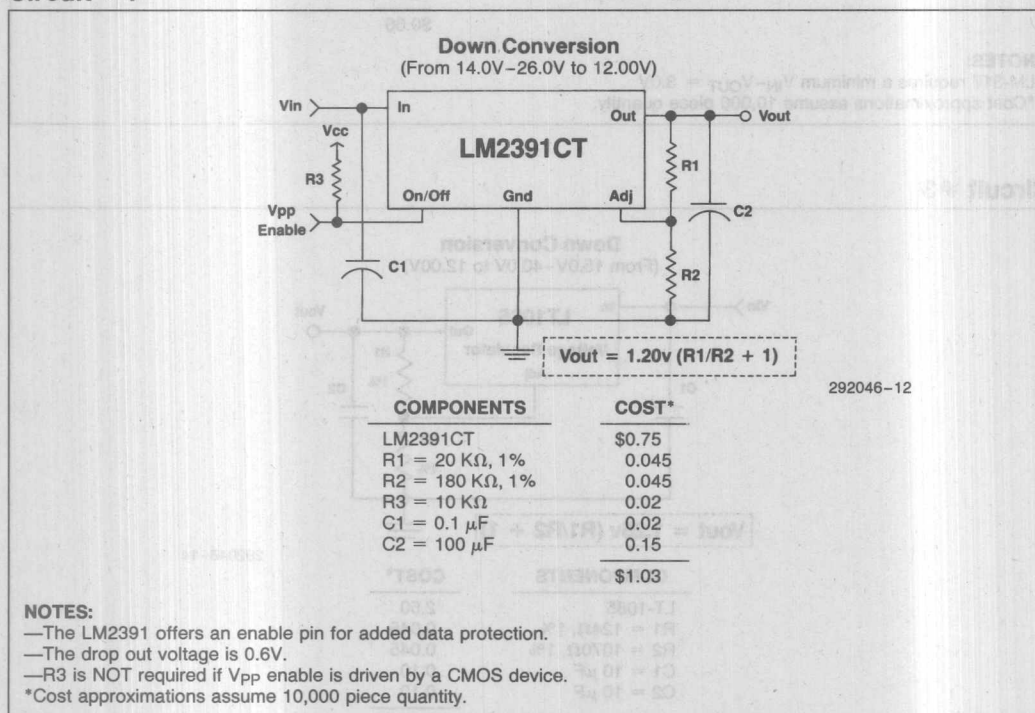
Circuit #4—5V to 12V Boost

Circuit #5—5V to 12V Boost

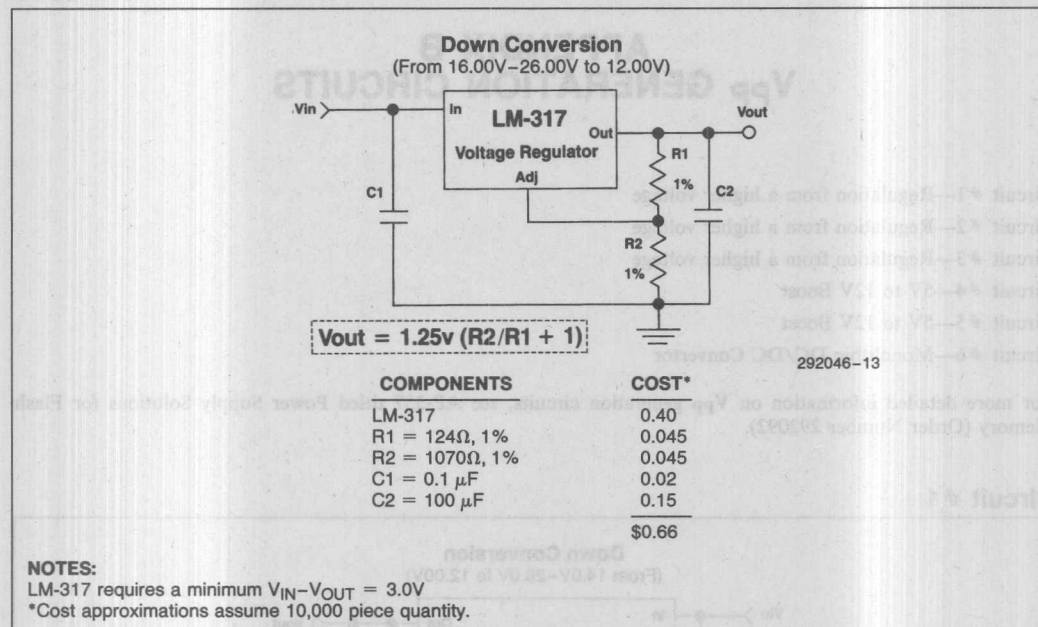
Circuit #6—Monolithic DC/DC Convertor

For more detailed information on V_{PP} generation circuits, see AP-357 titled Power Supply Solutions for Flash Memory (Order Number 292092).

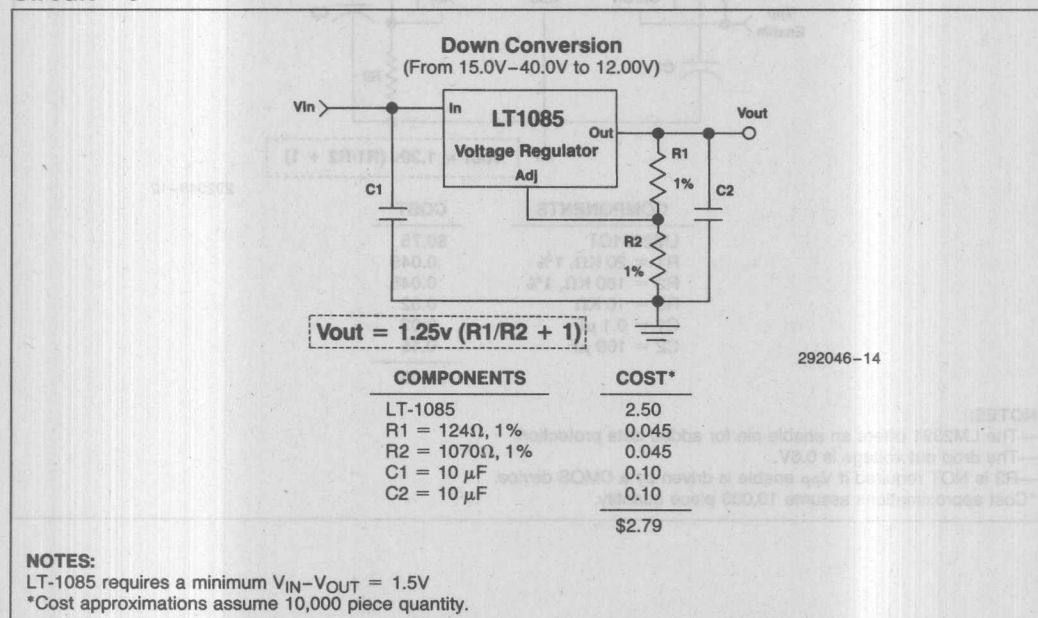
Circuit #1



Circuit #2

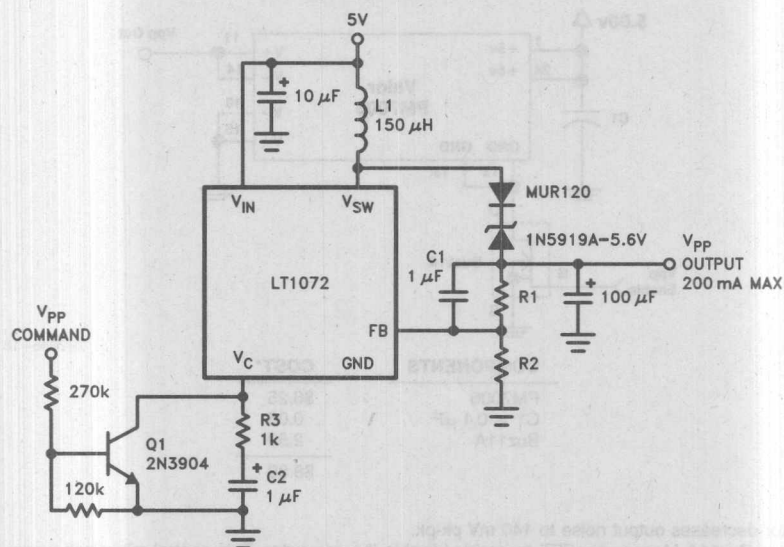


Circuit #3



Circuit #4

Up Conversion
(From 5V to 12.0V)



COMPONENTS

COST*

LT1072	1.82
R1 = 10.7k, 1%	0.045
R2 = 1.24k, 1%	0.045
R3 = 1k, 5%	0.02
R4 = 120k, 5%	0.02
R5 = 270k, 5%	0.02
C1 = 1 µF	0.10
C2 = 1 µF	0.10
C3 = 10 µF	0.15
L1 = 150 µH	1.00
Q1 = 2N3904	0.10

\$3.42

V _{PP} OUT	R1	R2	Resistor Tolerance
12.0V	10.7k	1.24k	1%

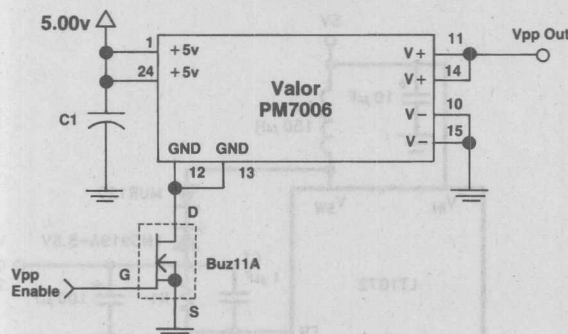
NOTES:

Drive V_{PP} COMMAND low to turn on the circuit.

*Cost approximations assume 10,000 piece quantity.

292046-33

Circuit # 5

Up Conversion Circuit
(From 5.0V to 12.0V)

COMPONENTS

COST*

PM7006	\$6.25
C1 = 0.1 μ F	0.05
Buz11A	2.59
	<hr/>
	\$8.89

NOTES:

1. The capacitor decreases output noise to 140 mV pk-pk.
2. We added the Buz11A Mospower nFET to enable/disable the converter. This control minimizes power consumption which under full load can reach 600 mA.
3. The voltage drop across the switch is 0.1V. Due to this drop the PM7006 will not maintain the Vpp spec with 10% fluctuations in VCC supply.

*Cost approximations assume 10,000 piece quantity.

APPENDIX C

LIST* OF DC-DC CONVERTER COMPANIES

AT&T MICROELECTRONICS†
3000 Skyline Drive
Mesquite, TX 75149
Tel: (800) 526-7819
Fax: (214) 284-2317

BURR-BROWN CORP.†
P.O. Box 11400
Tucson, AZ 85734
Tel: (800) 548-6132
Fax: (602) 741-3895

LINEAR TECHNOLOGY CORP.Δ
1630 McCarthy Blvd.
Milpitas, CA 95035
Tel: (408) 432-1900
Fax: (408) 434-0507

MAXIM INTEGRATED PRODUCTSΔ
120 San Gabriel Drive
Sunnyvale, CA 94086
Tel: (408) 737-7600
Fax: (408) 737-7194

MOTOROLA INC.Δ
2100 E. Elliot Rd.
Tempe, AZ 85284
Tel: (800) 845-6686

NATIONAL SEMICONDUCTOR CORP.Δ
Mt. Prospect, IL 60056
Tel: (800) 628-7364
Fax: (800) 888-5113

SHINDENGEN AMERICA, INC.†
2649 Townsgate Rd., Suite 200
Westlake Village, CA 91361
Tel: (800) 634-3654
Fax: (805) 373-3710

SILICONIX INC.Δ
2201 Laurelwood Rd.
Santa Clara, CA 95056
Tel: (800) 554-5565
Fax: (408) 727-5414

TOKO AMERICA, INC.†
1250 Feehanville Drive
Mount Prospect, IL 60056
Tel: (708) 297-0070
Fax: (708) 699-7864

VALOR ELECTRONICS†
6275 Nancy Ridge Dr.
San Diego, CA 92121
Tel: (619) 458-1471

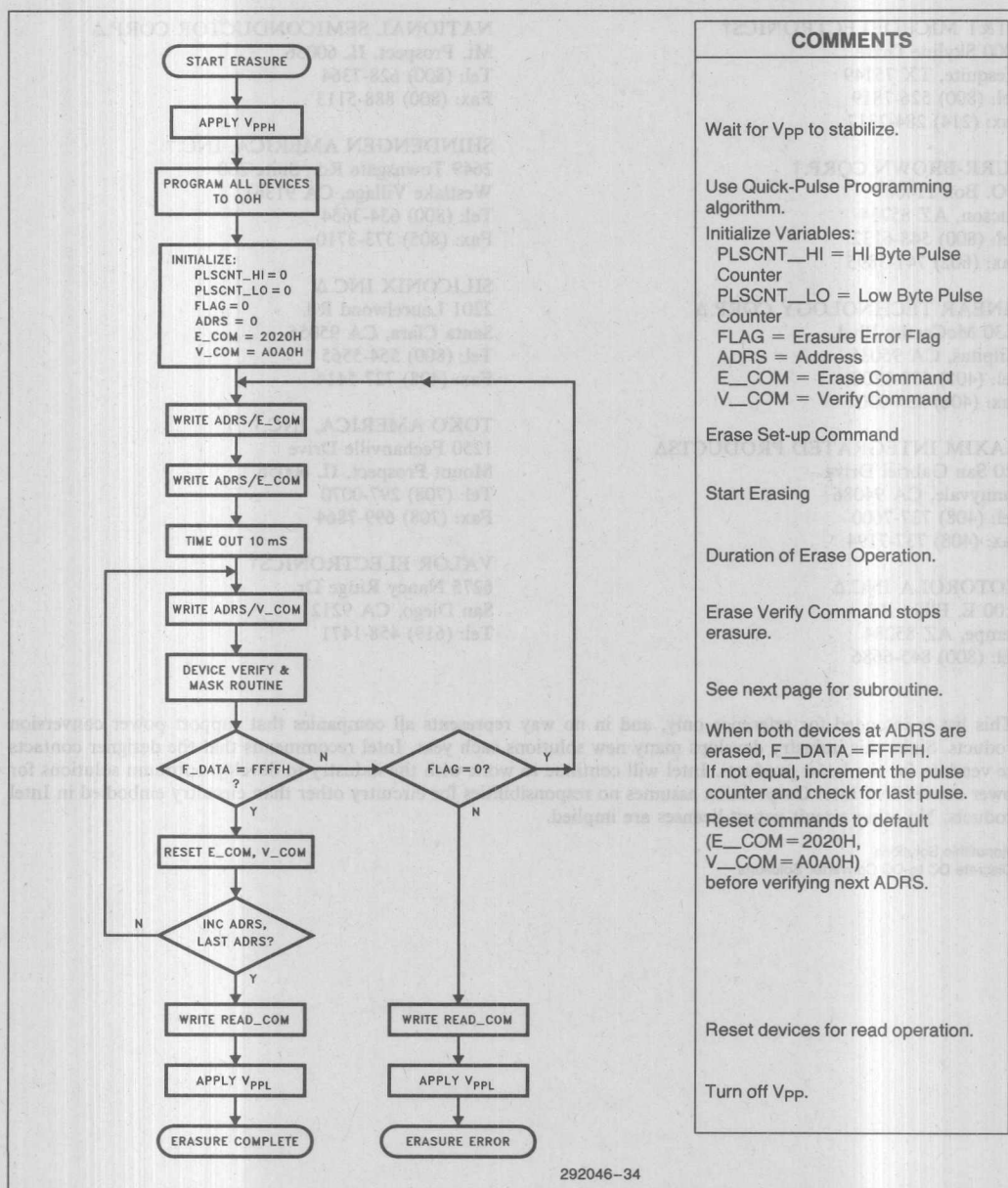
*This list is intended for reference only, and in no way represents all companies that support power conversion products. Since this industry develops many new solutions each year, Intel recommends that the designer contacts the vendors for the latest products. Intel will continue to work with the industry to develop optimum solutions for power conversion. Intel Corporation assumes no responsibilities for circuitry other than circuitry embodied in Intel products. No other circuit patent licenses are implied.

†Monolithic Solutions

ΔDiscrete DC to DC Converter Solutions

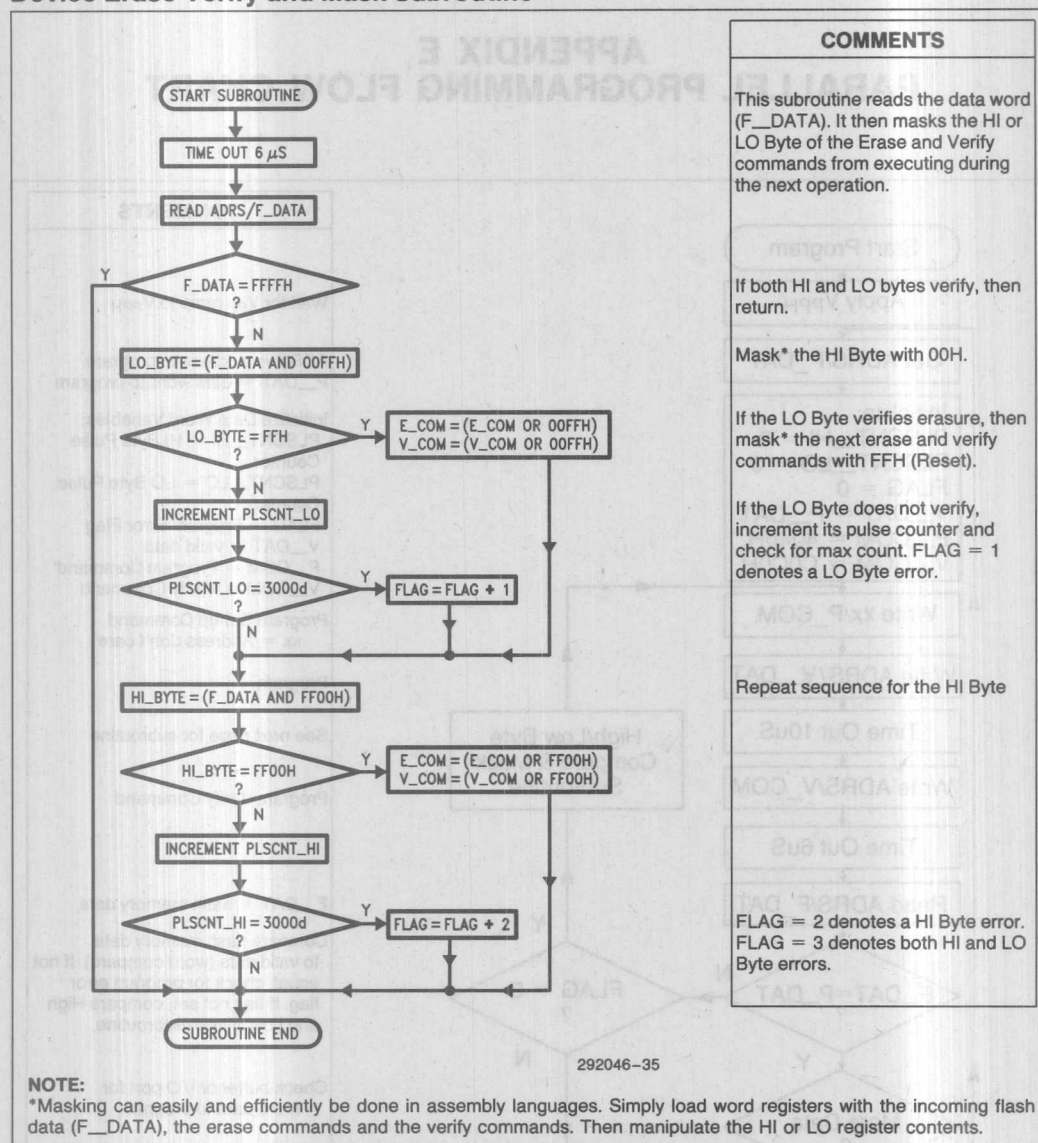
APPENDIX D

PARALLEL ERASE FLOW CHART

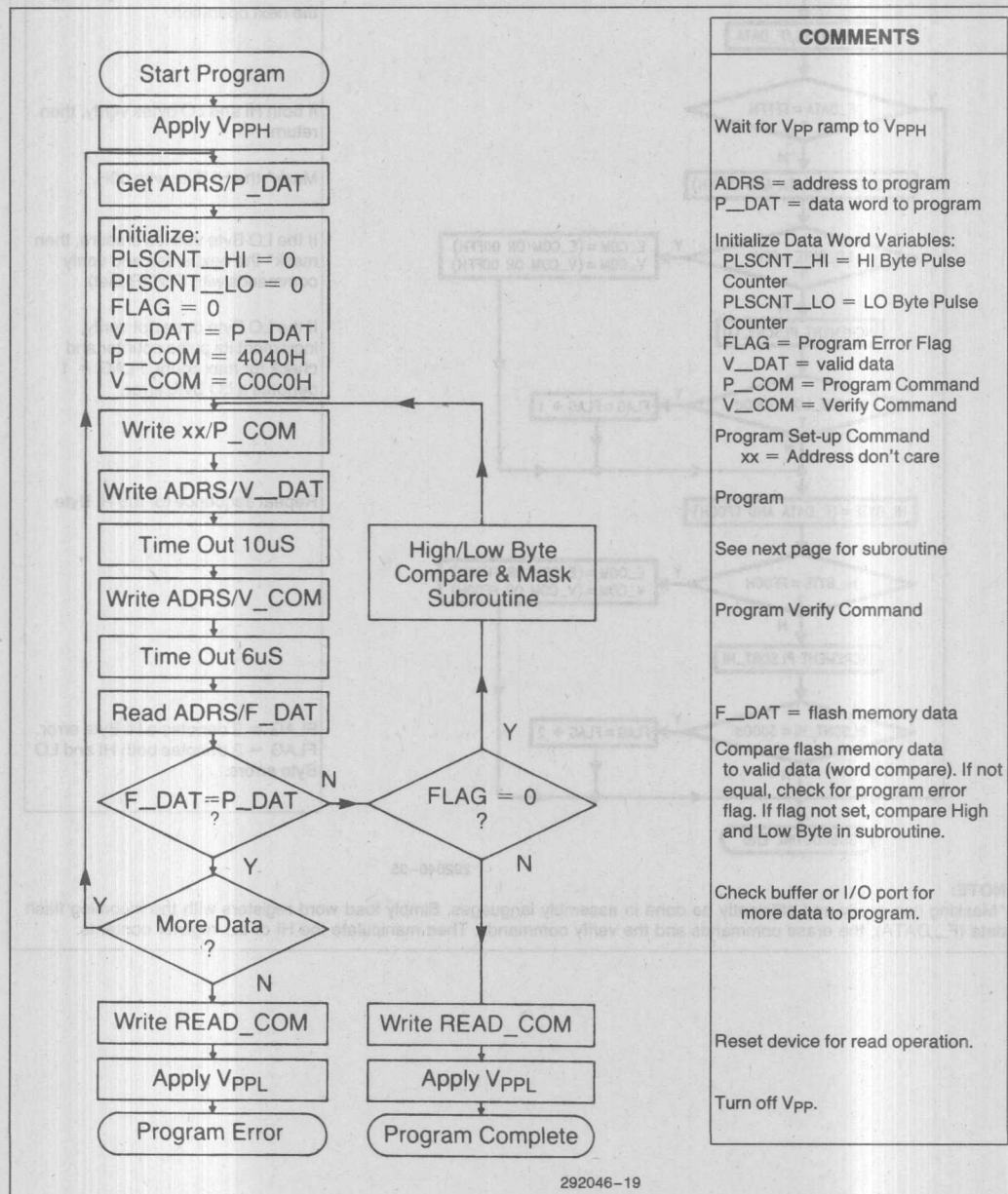


292046-34

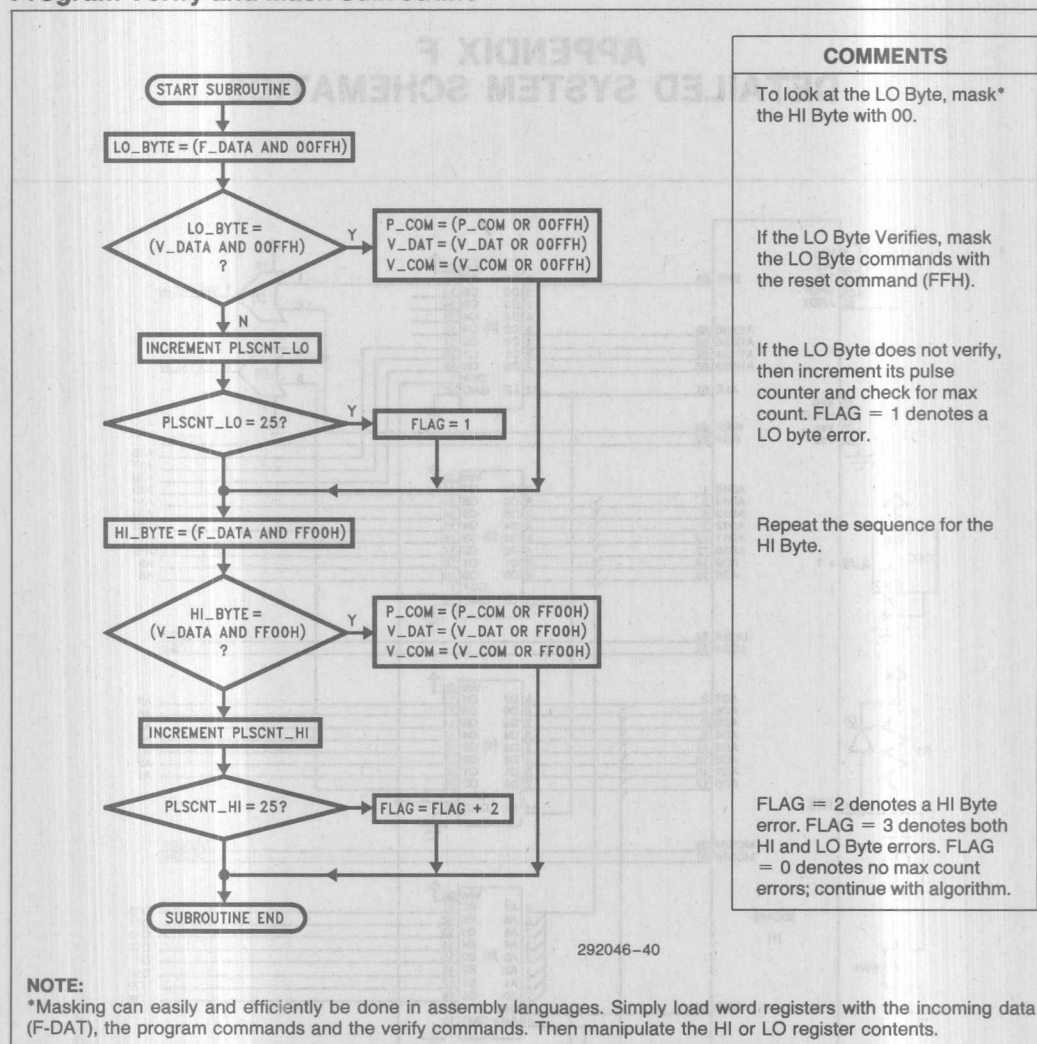
Device Erase Verify and Mask Subroutine



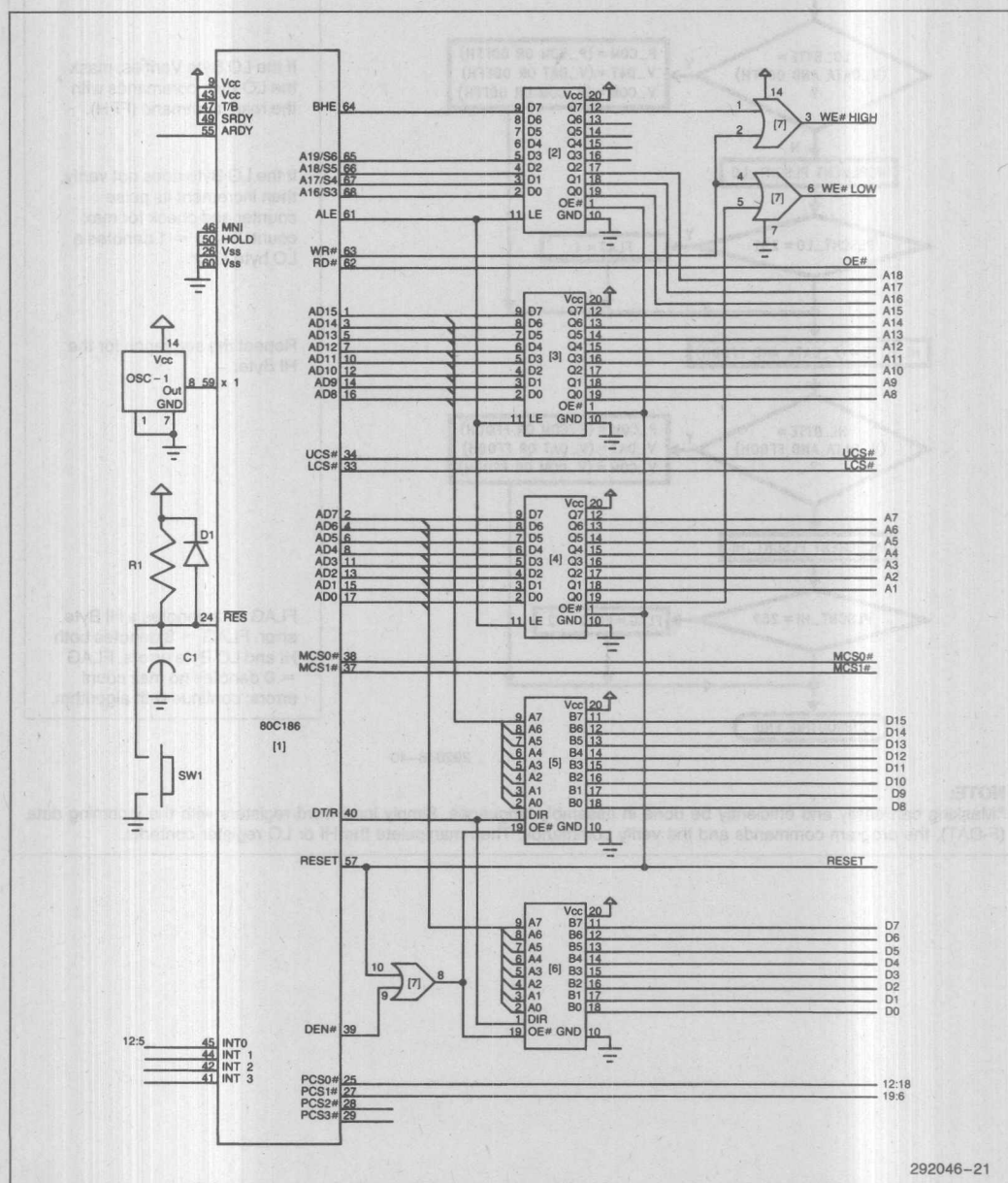
APPENDIX E PARALLEL PROGRAMMING FLOW CHART

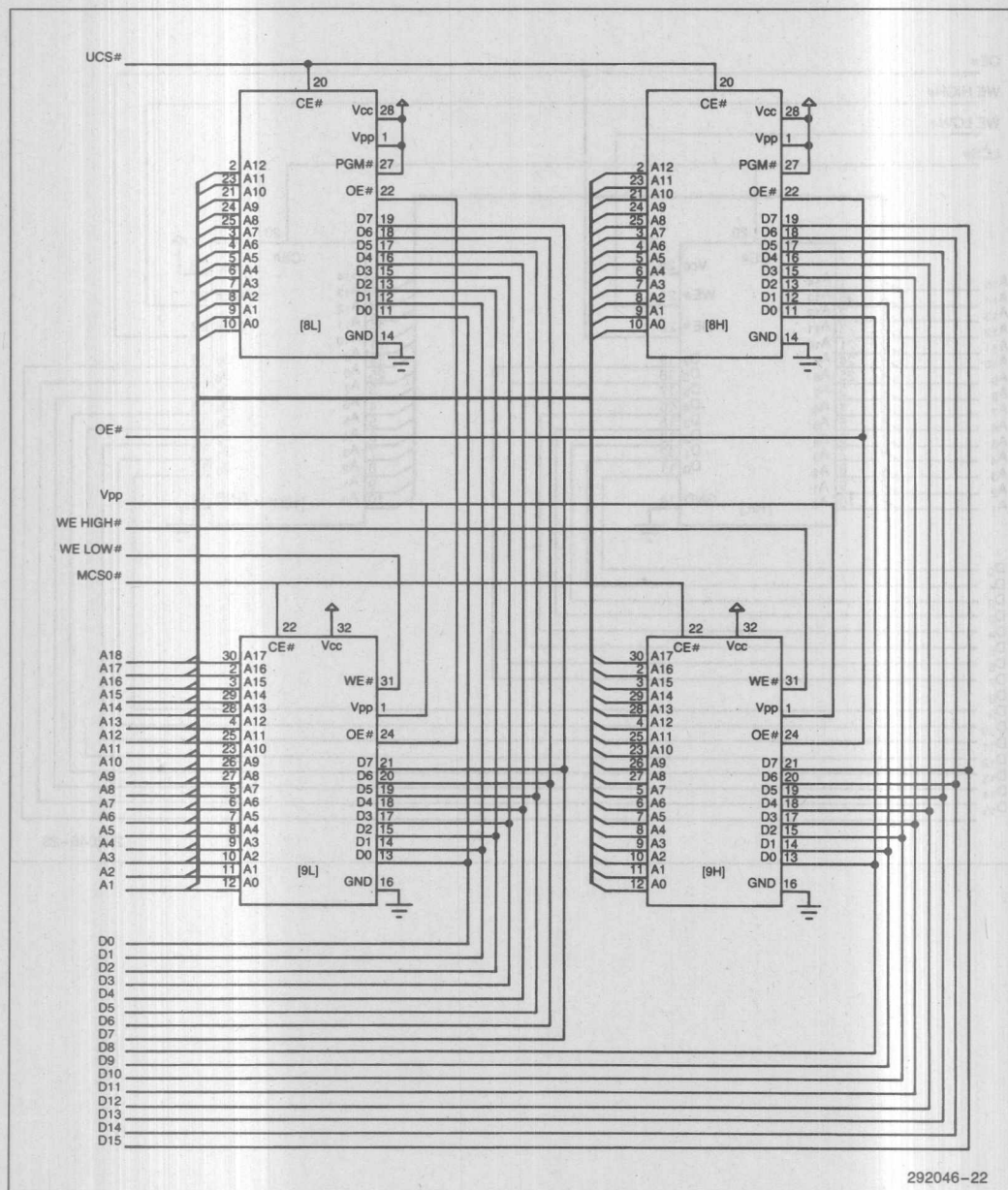


Program Verify and Mask Subroutine



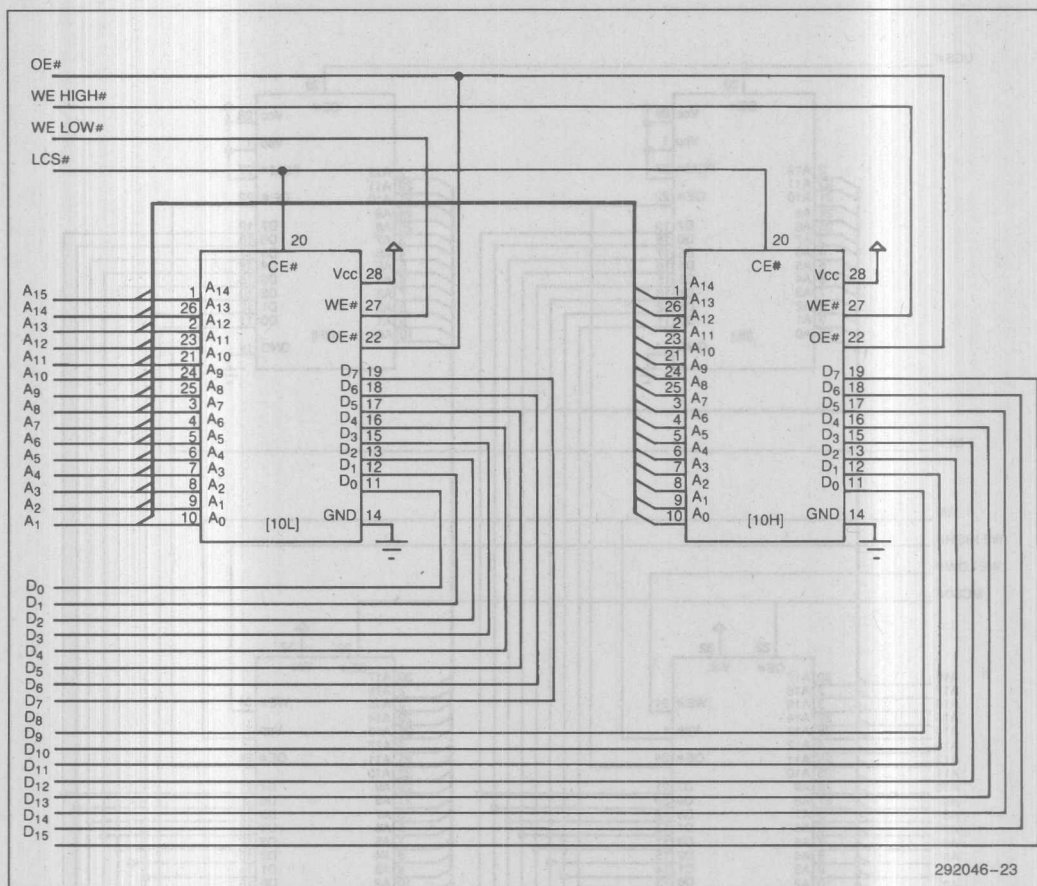
APPENDIX F DETAILED SYSTEM SCHEMATICS

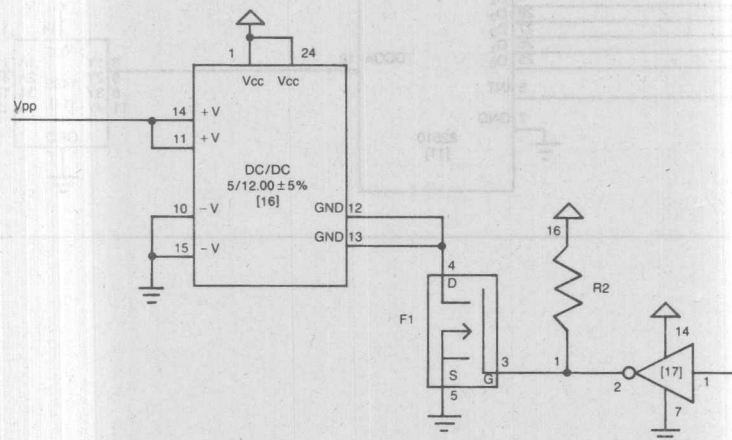
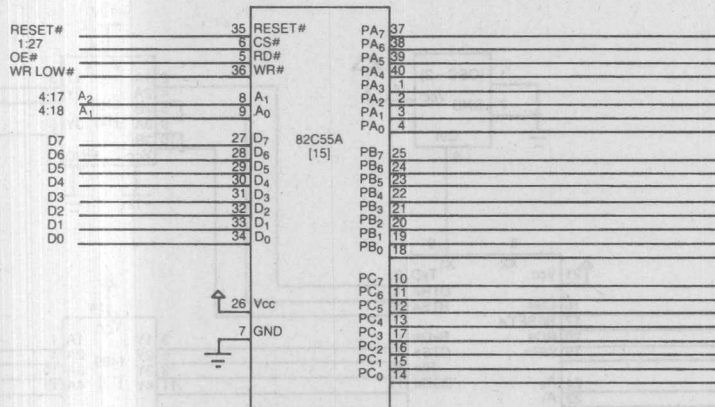




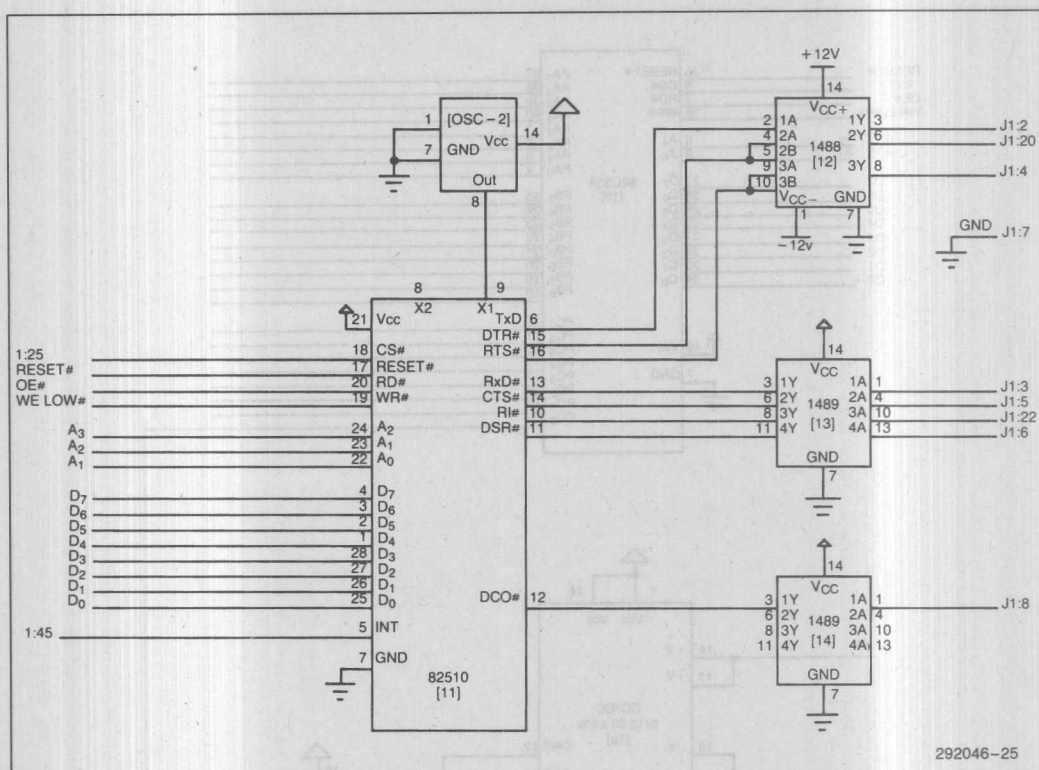
5

292046-22





292046-24



256K FLASH MEMORY DEMO PARTS LIST

Device	Component	Pins	Description
[1]	80C186	68	16-bit high integration CPU
[2,3,4]	74HC573	20	Latch
[5,6]	74HC245	20	Transceiver
[7]	74HC32	14	OR gate
[8L,8H]	27C64	28	16 Kbyte EPROM
[9L,9H]	28F256A	32	64 Kbyte flash memory
[10L,10H]	32K x 8 SRAM	28	64 Kbyte SRAM
[11]	82510	28	Asynchronous Serial Controller
[12]	14C88	14	RS-232 Line Driver
[13,14]	14C89	14	RS-232 Line Receiver
[15]	82C55A	40	Programmable Peripheral Controller
[16]	PM7006	24	DC/DC Converter (5V–12.00V)
[17]	7406	14	Inverter—Open Collector (O.C.)
C1	20 μ F	2	Capacitor for CPU reset
D1	1N914	2	Diode for CPU reset
F1	BUZ11A	3	MOSPOWER nFET
J1	DB-25	25	Connector (male)
OSC-1	20 MHz	14	CPU Oscillator
OSC-2	18.432 MHz	14	Serial Controller Oscillator
R1	10 K Ω	2	1/4W, 10% Resistor for CPU reset
R2	1 K Ω	2	1/4W, 10% Resistor for O.C. pull-up
SW1		3	Momentary Push Button for CPU reset

NOTES:

1. Place a 0.1 μ F bypass capacitor at the V_{CC} input of each IC.
2. Place a 0.1 μ F bypass capacitor on the V_{pp} input of each 28F256 flash memory.

28F512 UPGRADE FOR THE 80C186/FLASH MEMORY DESIGN

To upgrade the 80C186/Flash memory design to handle 28F512's, the range of the CE# signal has to be increased. There are a number of ways to generate a CE# signal that will span the 128 Kbyte address range of two 28F512 devices.

1. AND two of the current MCS lines together (defined for 64 Kbytes each); or

2. Change the MCS individual block-select size from 64 Kbytes:

MMCS_VALUE = 41F8H,
MPCS_VALUE = 0A0B8H

to 128 Kbytes:

MMCS_VALUE = 01FEH,
MPCS_VALUE = 0C0BEH

Also, cut the CE# trace to the RAM sockets. Then wire MCS0# to the RAM CE#. This eliminates the MCS0# and LMCS# range overlap caused by increasing the MCS range to 128 Kbytes. See 80C186 Data Sheet page 21 and 22 (Order # 270354).

28F010 UPGRADE TO THE 80C186/FLASH MEMORY DESIGN

To upgrade the 80C186/flash memory design to handle 28F010's, a CE# signal has to be generated. There are a number of ways to generate a CE# signal that will span the 256 Kbyte address range of two 28F010 devices.

1. AND two of the MCS lines together (defined for 128 Kbytes each as noted in the 28F512 modifications):

Cut the LMCS trace to the RAM sockets. Connect MCS0# to CE# on the RAM sockets (U10L,UH).

Cut the MCS2# trace to the flash memory. Add an AND gate. Connect MCS2# (cut trace) and MCS3# to the inputs of the AND gate. Then wire the AND gate output to the CE# of the flash memories.

Also, change the onboard memory MCS register to:

```
MMCS_VALUE=01FEH, MPSC_
VALUE=0C0BEH [128K blocks],
```

and delete:

```
LMCS_REG and LMCS_Value.
```

2. Add a decoder;

Add a decoder (74HC138). Connect address lines A18 and A19 to the B and C inputs of the decoder. Tie the A input of the decoder low, and enable all the enables. By using outputs Y0, Y2, Y4, and Y6, you have four CE# lines decoding 256 Kbyte blocks each.

Cut the MCS2# trace to the flash memories. Connect the Y2 output from the decoder to the CE# input of the flash memory.

3. Replace the address latch (U2) with a PLD that latches and decodes.

Program a 5C032 as an integrated latch and decoder. Replace the upper address latch [U2] with the Intel 5C032 EPLD. Cut the CE# trace to the flash memories. Connect the flash memories' CE# to the 5C032 pin 12. This maps the address space 40000H to 7FFFFH. See Figures 1 and 2 for a comparison of the 74HC573 (U2) and programmed 5C032 pin outs. Figure 3 is the source code for the EPLD.

Also, change the value of the MMCS and MPSC registers to 64 Kbyte blocks so that the MCS0# range does not overlap the LMCS range.

```
MMCS_VALUE=41F8H, MPSC_
VALUE=0A0B8H.
```

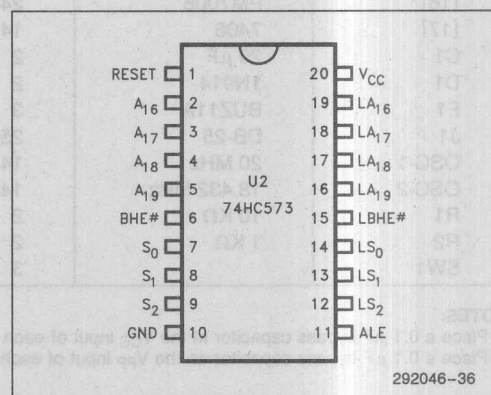


Figure 1. Latch Pinout

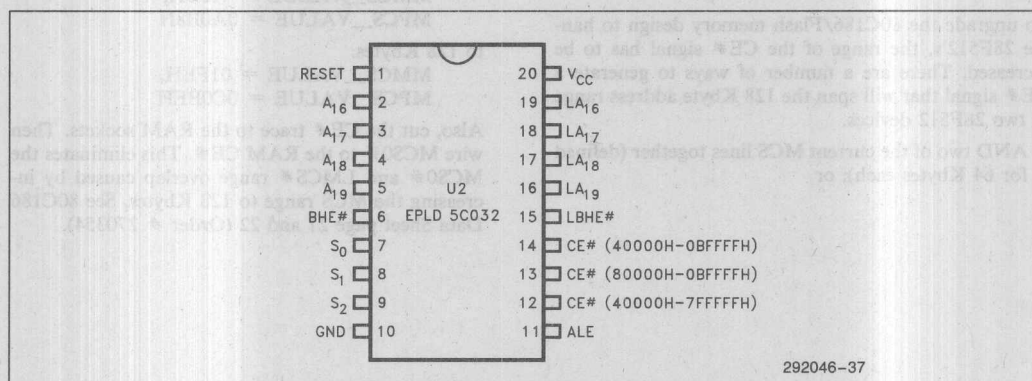


Figure 2. Integrated Latch and Decoder

Thom Bowns - PLFG Applications

Intel

January 13, 1989

EPLD HOTLINE: 1-800-323-EPLD

002

5C032

Custom Latched Decoder

OPTIONS: TURBO=ON

PART: 5C032

INPUTS: ALE@11, RESET@1, A19@5, A18@4, A17@3, A16@2, nBHE@6

OUTPUTS: LA18@17, LA17@18, LA16@19, LnbHE@15, nCE3@14, LA19@16,
nCE2@13, nCE1@12

NETWORK:

```

ALE = IN (ALE)
RESET = INP (RESET)
nRESET = NOT (RESET)
A19 = INP (A19)
A18 = INP (A18)
A17 = INP (A17)
A16 = INP (A16)
nBHE = INP (nBHE)
LA19, LA19 = COIF (LA19d, nRESET)
LA18, LA18 = COIF (LA18d, nRESET)
LA17, LA17 = COIF (LA17d, nRESET)
LA16, LA16 = COIF (LA16d, nRESET)
LnbHE, LnbHE = COIF (LnbHE, nRESET)
nCE3, nCE3 = COIF (nCE3, nRESET)
nCE2, nCE2 = COIF (nCE2, nRESET)
nCE1, nCE1 = COIF (nCE1, nRESET)

```

EQUATIONS:

```

LA19d = A19 * ALE + LA19 * !ALE;
LA18d = A18 * ALE + LA18 * !ALE;
LA17d = A17 * ALE + LA17 * !ALE;
LA16d = A16 * ALE + LA16 * !ALE;
LnbHEd = nBHE * ALE + LnbHE * !ALE;
nCE3d = nCE3EQN * ALE + nCE3 * !ALE;
nCE2d = nCE2EQN * ALE + nCE2 * !ALE;
nCE1d = nCE1EQN * ALE + nCE1 * !ALE;
nCE2EQN = !(A19 * !A18);
nCE1EQN = !(A19 * A18);
nCE3EQN = !(A19 * A18 + A19 * !A18);

```

END\$

Figure 3. Source Code for the Integrated Latch and Decoder

APPLICATION NOTE

Guide to First Generation Flash Memory Reprogramming

APPLICATIONS ENGINEERING STAFF

October 1993

CONTENTS

PAGE

INTRODUCTION TO REPROGRAMMING	5-164
You Are in Control	5-164
FUNDAMENTALS OF FLASH OPERATION	5-164
Adaptive vs. Brute Force Algorithms	5-164
Moving Charge & Other Factors You Should Know	5-165
ERASURE—THE GOLDEN RULE	5-168
Margin for Error	5-168
Most Common Development Issues	5-169
Device Initialization and Reset	5-169
The Erase Algorithm Interpreted	5-171
The Program Algorithm Illuminated	5-173
Ramifications of the Golden Rule	5-173

CONTENTS

PAGE

DEBUGGING YOUR CODE AND OTHER TIPS ON TESTING	5-174
Software Drivers Save You Time	5-174
Timers, Test Loops and Assembly Level Programming	5-174
Programming—The Key to Proper Erasure	5-174
16- and 32-Bit Systems	5-175
Logic Analyzers and In-Circuit Emulators	5-175
Testing Your Software—One More Time	5-175
Watchdog Timer Debug Circuit	5-176
TROUBLE SHOOTING GUIDE	5-177
Determining the Root Cause	5-177

INTRODUCTION TO REPROGRAMMING

You Are in Control

Rewriting any type of memory requires hardware or software control. Traditional EEPROM designers combined all control functions into each chip's periphery. This provided a highly functional chip but at a high price. On the other hand, DRAM designers provided a bulk memory with little integrated peripheral circuitry. Each system designer then accommodated the DRAM with external refresh signals and learned quickly that failure to refresh yielded non-functioning memory boards. Initially, software drivers controlled DRAM refresh; today controllers provide the same function.

Similarly, early disk drives required every user to write software to manipulate drive head movement. Failure to follow drive specifications and algorithms caused irreversible head crashes. Leading-edge engineers faced these challenges and triumphed, as evidenced by the sophisticated systems available today.

Since 1988, thousands of engineers have written software to direct flash memory reprogramming. With first generation flash memories, one sends a control signal to a device to begin and end programming or erasure. It is a simple process implemented on more than 40 million units, however care must be taken. If algorithms are not properly followed, a device may be rendered inoperable. This document discusses proper software and debug technique, which yields dependable first generation flash memory operation. First generation products include the 28F256A, 28F512, 28F010 and 28F020. All second and third generation Intel Flash Memories contain automated program and erase routines.

FUNDAMENTALS OF FLASH MEMORY OPERATION

Adaptive vs Brute Force Algorithms

Many designers use EPROMs regularly. Few consider the programming algorithms because the PROM programmer vendors take care of that function.

Two types of algorithms are in use today:

- Adaptive Algorithms
- Brute Force Algorithms

Adaptive algorithms such as Intel's Quick-Pulse Programming and Quick-Erase algorithms reduce programming time. A feedback mechanism recognizes when each byte has been programmed sufficiently. You may ask how is the point of sufficiency determined?

One simply adds the net effects of V_{CC} and temperature variations, and superimposes on those factors the normal EPROM charge leakage to obtain the answer. The next question is how can these factors be checked?

NOTE:

EPROM and EEPROM charge leakage occurs over a very long time—typically 100 years. Reliability papers often discuss charge leakage in terms of the memory's data retention characteristic.

If you look at EPROM programming algorithms, you will notice that V_{CC} is elevated during programming. The elevated V_{CC} acts as the feedback mechanism for the adaptive algorithm. Reading the device and checking for program completion is called verification, or margining. (One is checking the margin to V_{CC} fluctuations.) For example, if the part can be verified at 6.25V, then it can withstand the fluctuations and normal charge leakage.

During the past few years most major EPROM manufacturers have converted to adaptive algorithms. The algorithm loops back and programs a byte again if the first program operation does not verify at the elevated voltage.

Brute force algorithms simply program each byte multiple times, typically with long program durations. This type of algorithm has no in-system margin verification. That is they *assume but never verify* program margin to the typical environment effects.

Many flash memories that specify a brute force algorithm may fail to retain data for 10 years. Additionally, they may not read the data correctly even at specified V_{CC} and temperature extremes.

Intel's flash memory program and erase algorithms are both adaptive. They offer margin verification without requiring users to elevate V_{CC} in-system. When issued a command to program verify, the memory's command register logic taps an internally-generated elevated V_{CC} from the user-supplied external V_{pp} (12V). This is why it is essential that you provide the specified V_{pp} voltage and follow the given adaptive algorithms. Intel's adaptive algorithms, combined with the command register architecture, assures reliable code and data storage and dependable system operation.

Figure 1 shows an example of an adaptive byte programming algorithm. Appendix A compares the algorithm in Figure 1 to a brute force approach.

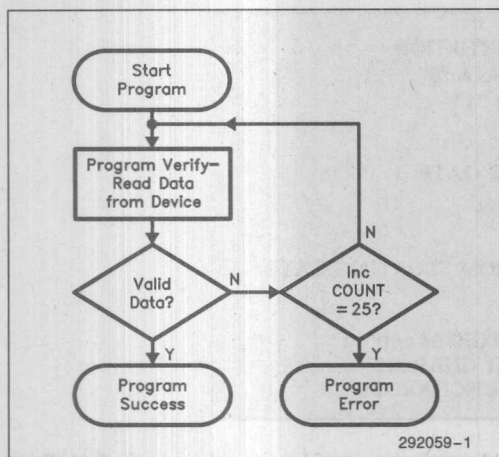


Figure 1. The flow chart shows the fundamental nature of an adaptive algorithm. Based on the outcome of program verification, the flow may loop back for another program operation.

Moving Charge and Other Factors You Should Know

This section discusses the mechanics of flash memory programming. For most system designers, transistor-

level discussions were last heard in college. We may recall that DRAM consists of a storage capacitor and a transistor. We remember this clearly because failure to refresh that capacitor causes systems to malfunction. In like fashion, one should understand the fundamentals of flash memory reprogramming. The understanding will enable error-free memory operation and reliable system performance.

In simplest terms, each data bit equates to a memory cell. Intel's flash memory uses one transistor per cell with the smallest possible architecture. This delivers the lowest cost per bit and highest capacity, leveraging system software (rather than bulky, complex cells) for reprogramming control.

Figure 2 shows a simplified cross section of Intel's flash memory transistor. Note the structure; the cell is a stacked gate MOS transistor. An isolated floating gate stores the memory charge. The floating gate consists of a layer of (conductive) polysilicon surrounded by (non-conductive) oxide layers.

On a DRAM cell, each transistor connects to a capacitor which stores the memory charge. The major difference between flash memory and DRAM derives from their cell structure. The DRAM cell loses its charge if not refreshed within a few milliseconds. On the other hand, the flash memory floating gate maintains its charge for typically 100 years. The structure is isolated and insulated by the field and gate oxides—hence the name “floating” gate.

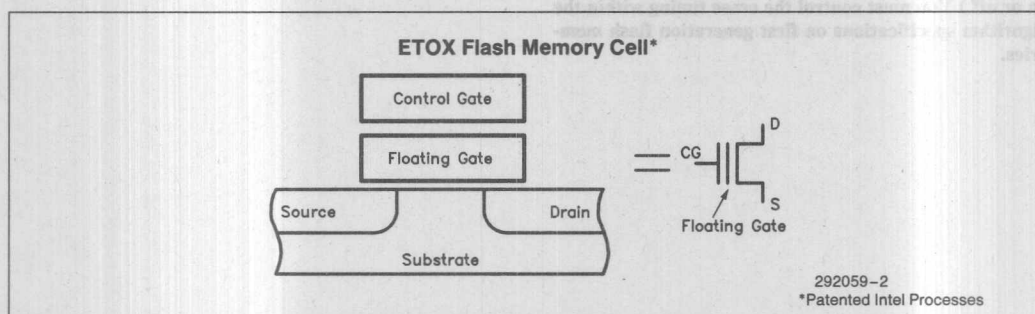


Figure 2. Simplicity of design assures increasing densities, manufacturability and reliability. These are the attributes that drive mainstream memories.

CONTRARY TO INTUITION
CHARGE = DATA "0"
NOT DATA "1"

PROGRAMMING:
 ADDS CHARGE TO FLOATING GATE
 → DATA = 0

ERASURE:
 REMOVES SOME CHARGE FROM FLOATING GATE
 → DATA = 1

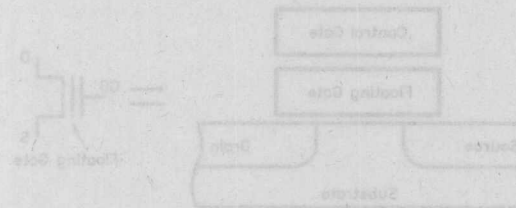
PROGRAMMING DATA WITH MIXED 0s and 1s:
 → ONLY DATA "0" BITS GET CHARGED
 → DATA "1" BITS REMAIN UNCHARGED

Changing the memory contents is simple. Figure 3 shows two memory cells—one being erased and one being programmed. Erasure removes charge from all bits simultaneously. Programming adds charge to selected bits. During erasure, not all charge is removed. The erase verify operation tells the system when enough charge has been removed. At that point, the flash memory behaves like a U.V.-erased EPROM.

Removing too much charge by erasing too long renders the memory unprogrammable. Excessive erasure lowers the cell threshold to the point where the transistor is always on and always reads data "1". (Recall that the cell threshold, V_t , determines when the transistor turns on or off.) You must control the erase timing within the algorithm specifications on first generation flash memories.

A second erase consideration relates to the first. Prior to erasing the chip, you must blanket program all bytes to data 00h, regardless of the previous data. This step equalizes the charge on all transistors.

If you skip this step and proceed directly to erasure, an interesting thing happens. Consider a typical byte programmed with data 0AAh (1010 1010b). While programming this data, bits with data "1" remain erased (charge removed), and bits with data "0" are programmed (charged added). Following programming, normal read operations sense whether a memory transistor has more or less charge and drives the outputs accordingly.



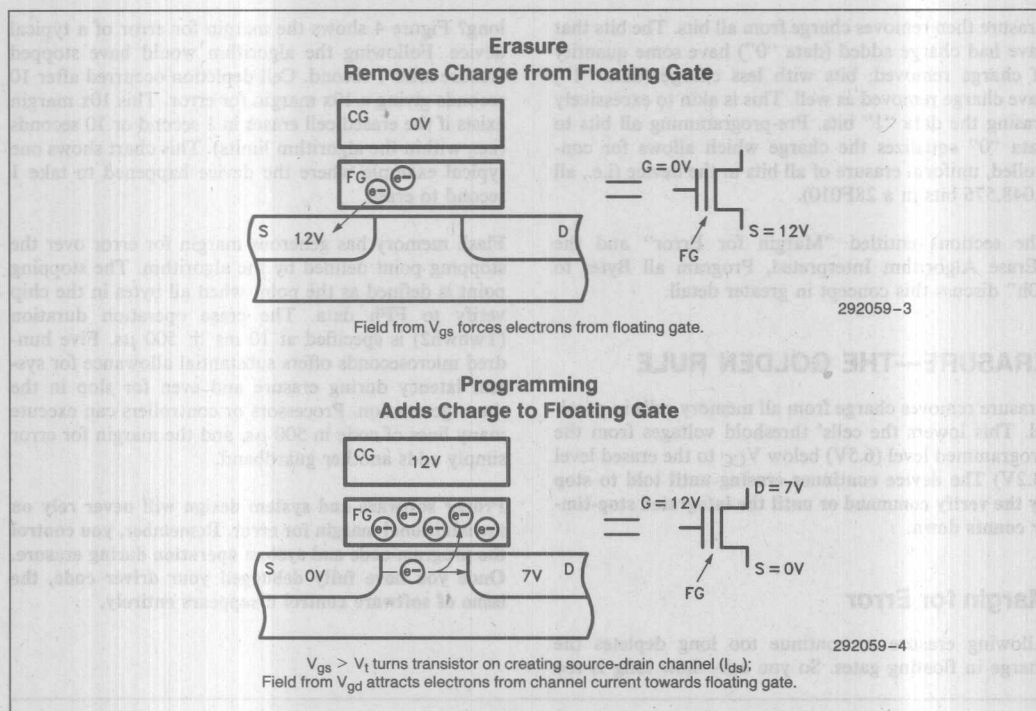


Figure 3. Flash memory cells during erasure and programming. Note the movement of charge on and off the floating gate. The charge adjusts the cell threshold, which tells the outputs whether a bit (transistor) is on or off.

Erase then removes charge from all bits. The bits that have had charge added (data "0") have some quantity of charge removed; bits with less charge (data "1") have charge removed as well. This is akin to excessively erasing the data "1" bits. Pre-programming all bits to data "0" equalizes the charge which allows for controlled, uniform erasure of all bits in the device (i.e., all 1,048,576 bits in a 28F010).

The sections entitled "Margin for Error" and the "Erase Algorithm Interpreted, Program all Bytes to 00h" discuss this concept in greater detail.

ERASURE—THE GOLDEN RULE

Erase removes charge from all memory cells in parallel. This lowers the cells' threshold voltages from the programmed level (6.5V) below V_{CC} to the erased level (3.2V). The device continues erasing until told to stop by the verify command or until the integrated stop-timer counts down.

Margin for Error

Allowing erasure to continue too long depletes the charge in floating gates. So you ask—how long is too

long? Figure 4 shows the margin for error of a typical device. Following the algorithm would have stopped erasure after 1 second. Cell depletion occurred after 10 seconds giving a 10x margin for error. This 10x margin exists if the erased cell erases in 1 second or 10 seconds (i.e., within the algorithm limits). This chart shows one typical example where the device happened to take 1 second to erase.

Flash memory has generous margin for error over the stopping point defined by the algorithm. The stopping point is defined as the point when all bytes in the chip verify to FFh data. The erase operation duration (Twhwh2) is specified at $10 \text{ ms} \pm 500 \mu\text{s}$. Five hundred microseconds offers substantial allowance for system latency during erasure and even for slop in the timer generation. Processors or controllers can execute many lines of code in $500 \mu\text{s}$, and the margin for error simply adds another guardband.

Proper software and system design will never rely on the additional margin for error. Remember, you control the program code and system operation during erasure. Once you have fully debugged your driver code, the issue of software control disappears entirely.

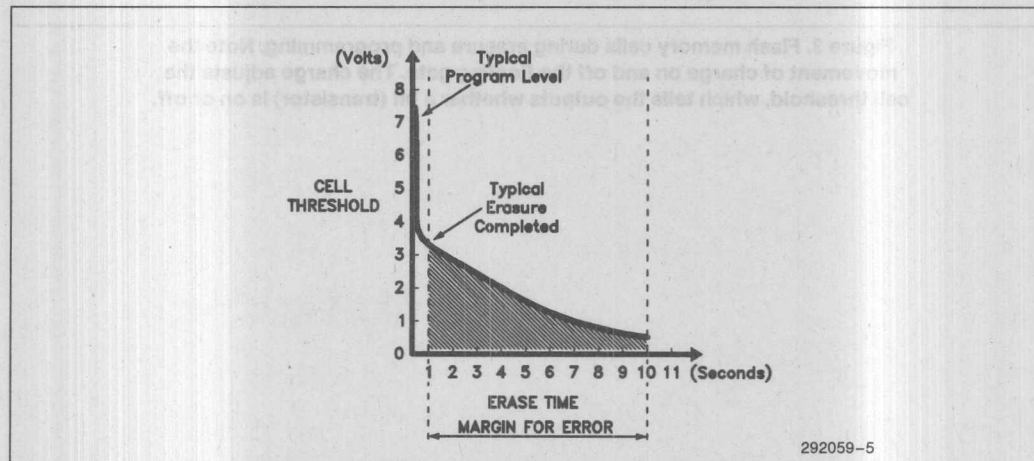


Figure 4. The logarithmic-decaying nature of erasure allows for 10x error in erase time before a device becomes inoperable. Remember, each device has its own erase time, thus the use of an adaptive algorithm.

Most Common Development Issues

Having covered the fundamentals of flash memory reprogramming, let's move on to the system's hardware and software perspectives. The following list of questions might have occurred to you . . .

- You have defined a system power supply with regulated 5V and 12V outputs (V_{CC} and V_{PP}). Due to the smaller capacitive load on the V_{PP} supply, V_{PP} powers up much faster than the V_{CC} supply. Will this affect the device?
- How does the flash command register architecture reset?
- Suppose your code sends a signal to start erasure, and never tells it to stop?
- Suppose your software delay timers are not calibrated. Instead of stopping erasure after ten milliseconds, the code issues the stop command after 10 seconds?
- Suppose your 6 μ s timer used between the erase and erase verify modes is only 2 μ s?
- Suppose you decide to skip the first erase operation (program all bytes to zero) because the device is already programmed with data?
- Suppose you are programming and erasing devices in a 16- or 32-bit system?

The answer to all these questions can be found in the following sections. The questions and the reasons all relate to the discussion of how the cell works.

Device Initialization and Reset

Many logic devices which contain command or control registers also have a reset pin. This pin serves two purposes: it resets the device's internal logic; and it synchronizes the device's clock to the system clock.

Intel's first generation flash memory command register and reprogramming circuitry **reset to the read mode** by three means:

1. raising or lowering V_{PP} with $V_{CC} = 5V$;
2. raising V_{CC} with $V_{PP} = 12V$;
3. issuing the reset command twice in succession.

NOTE:

Method 3 stops erasure or programming as well as resets the chip.

A few cases require closer consideration.

Case 1. The System Controls V_{PP} with a Switch

Assuming V_{CC} is stable with V_{PP} switches on, then the command register defaults to the read mode. No power-on reset is required.

Designers might opt to include the V_{PP} switch for either (or both) of two reasons. The first reason is power minimization. Depending on the technology used, a voltage regulator or pump's efficiency can range from 40%–85%. Switching off the V_{PP} supply minimizes system power consumption. See Appendix B for an example V_{PP} generation circuit with ON/OFF control capability.

The second reason is absolute data protection. This feature is not available to 5V-only EEPROM because the reprogramming voltages are generated internally. On that class of memory device, logic glitches can spuriously change data during system power up or power down. Flash memory's 12V power requirement offers absolute control over these concerns; with V_{PP} below $V_{CC} + 2V$, data protection is guaranteed. Internally, the electric fields are simply too weak to spuriously write data.

Case 2. V_{PP} Powers Up before V_{CC}

Systems with V_{PP} hardwired to a regulated transformer might encounter this case. Typically, V_{CC} will charge many more bypass capacitors than V_{PP} . V_{CC} will therefore power up much more slowly.

The flash memory power-down ($V_{CC} = 0V$) default state blocks V_{PP} from disturbing the array. These conditions hold while V_{CC} is below $\sim 2V$. Once V_{CC} rises above $\sim 2V$, the internal logic kicks in and resets the device to the read mode. (This is analogous to the internal V_{PP} reset condition described in Case 1.)

Should the three control pins glitch during the power-up phase (\overline{CE} low, \overline{WE} low, and \overline{OE} high), then the command register acts to filter the data. The command port will only react to the correct command sequence.

Designers might opt to hardwire V_{PP} for a number of reasons. The first reason is cost minimization. A regulated 12V secondary from a transformer is commonly available. Adding a switch or a power supply sequencer adds cost and complexity. The second reason involves consideration of the end application. Using the flash memory as a read/write memory requires optimization for the write cycle. Powering V_{PP} on before each write would waste considerable time.

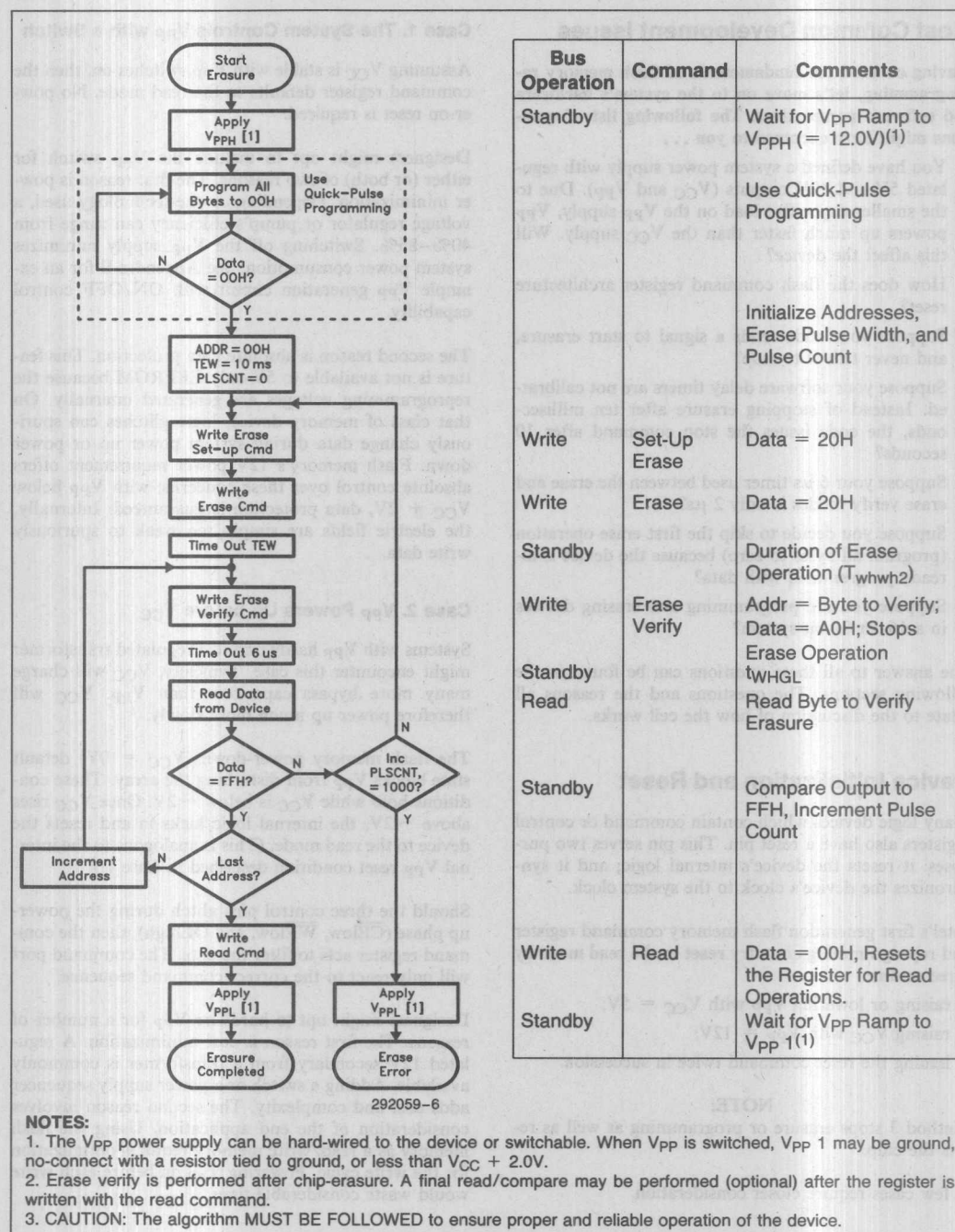


Figure 5. Quick-Erase Algorithm for Intel's First Generation Product Line

Case 3. Warm Resets

Warm resets, where the system maintains power while rebooting, requires closer inspection. Consider the situation where the system is reprogramming the flash memory and a hardware or software reset occurs.

The boot software would not realize that programming or erasure is ongoing and would not know to stop the reprogramming operation. Therefore safeguard against this condition with one of two means: 1) ensure that control logic switches V_{pp} off during reset; or 2) reset the flash memory before resetting the processor. For a software reset, simply add the flash memory reset command to the interrupt sequence. For hardware resets, wire the reset switch to the interrupt controller instead of directly to the reset input. Hardware resets would then execute the software interrupt sequence. Intel's second and third generation flash memories all include a H/W reset pin (RP#) formally called PWD#. This pin is essential for any device with automation or embedded algorithms.

The Erase Algorithm Interpreted

The following section offers a block by block explanation of the Quick-Erase algorithm shown in Figure 5. Understanding the reasons behind a function will enable you to appreciate the importance of following the algorithm explicitly. Deviations will negatively affect the part's performance and should not be attempted. Note: the effect may not be immediately apparent.

Apply V_{ppH} (Optional, see Discussion on Device Initialization)

Switch on the local V_{pp} supply prior to erasure and programming. The time required for V_{pp} to reach its steady state $12 \pm 0.6V$ depends on the capacitive load and the impedance of the printed circuit board trace. If you measure this delay on a wire-wrapped prototype system, remember that temperature, printed circuit traces and the board's layout change the load seen by the V_{pp} generator. Allow V_{pp} sufficient time to ramp before proceeding with the next step.

Program All Bytes to 00h \rightarrow Data = 00h?

Prior to erasure, blanket program all addresses in the flash memory to 00h (charge state), regardless of the previous data. Verify that each address equals 00h before proceeding to the next address. If you use only part of the memory array, you still need to pre-program the entire array for erasure. An example where this is an issue is using a 512K in a 256K socket. A second example is a system where internal microcontroller memory overlaps the external flash memory space.

Programming data 00h equalizes the charge on every bit in the array. This is necessary because erasure removes charge from all cells regardless of their previous state.

For example, reconsider the byte containing AAh data (1010 1010b). If you skip the pre-program step, then during erasure when the data "0" bits get charge removed, the previously erased bits (data "1") lose additional charge. This drives the cell threshold a little lower. The next time you erase the chip and change the code, the threshold will drop to 2.8V.

If the memory transistor is not pre-programmed to data "0" before the next erasure, then its threshold will drop on successive reprogramming cycles (denoted by E3, E4, etc. in Figure 6). Repeated violations of the blanket programming requirement drives the threshold to the point where the transistor is stuck on (data = "1").

Variable Initialization

Initialize two variables and a constant: ADDR (address), PLSCNT (pulse count), and TEW (erase pulse width). The pulse count increments from 0 to a maximum of 1000 erase tries. The erase pulse width remains constant at 10 ms. The address increments from the flash memory starting address to the ending address during verification.

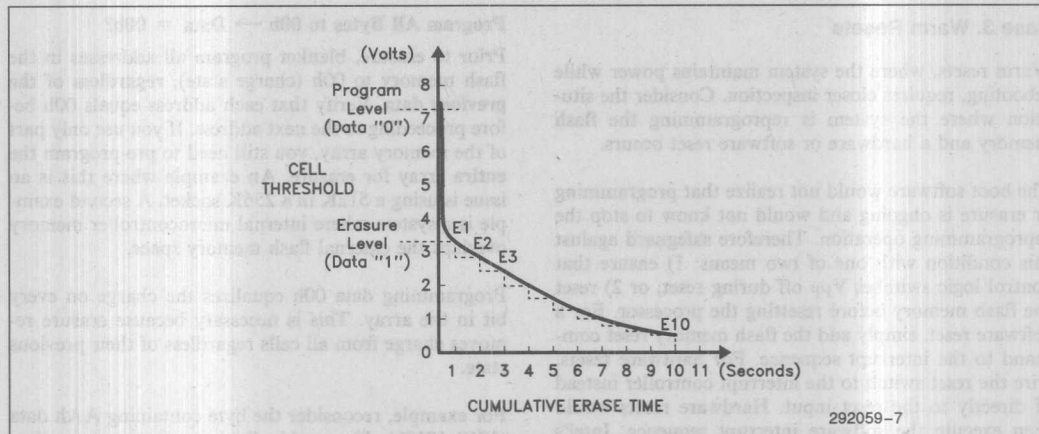


Figure 6. Successful erasure requires blanket programming all bytes to the data "0" level first. This prevents threshold decline on successive erase cycles (E2, E3, etc.). Very low thresholds cause the chip to malfunction.

Write Erase Set-Up Command

Write the erase set-up command (20h) to any flash memory address. This prepares the selected device for erasure, but does not activate the process. A second erase command (20h) is required. Any other data written to the flash memory between the set-up and erase commands will abort the sequence. Once the process is started, it will not stop until told to do so. The correct stop erase command is Erase Verify (A0h). However, any command including the Reset command is an illegal sequence and will stop erasure as well.

Write Erase Command

The erase command starts the erase process. Internally, the device switches the voltages on all memory cell drains, gates, and sources to the erase configuration.

Time Out Tew (10 ms)

Start your software or hardware timer. Until commanded to verify or until the integrated stop-timer counts down, the flash memory continues the erase process. Therefore, assign a high priority to the timer interrupt. If a higher priority interrupt occurs, stop the erase process and switch contexts (store all variables, registers, etc.). This will allow reentry into the erase procedure in a controlled fashion.

Write Erase Verify Command

Write the erase verify command (A0h) to the flash memory at the address given by the ADDR variable.

The erase verify command performs many tasks. Internally, the device stops erasure and latches the given address for verification. Additionally, the command changes the voltages on the memory cell drains, gates and sources to the erase verify configuration.

Time Out 6 μ s

This time out accounts for the internal slew rate of switching the memory array from the erase to the erase verify configuration. Do not attempt to read from the device before 6 μ s has passed; the device will appear to still be programmed. This is because you have not allowed sufficient time for the memory to change configurations. Your code will then interpret this as a need for extra erase operations, and will continue erasing the device.

NOTE:

6 μ s is a minimum specification. You can use the 10 μ s timer developed for the programming algorithm.

Read Data from Device

Read the data at the address given by the ADDR variable. This should be the same address driven with the erase verify command.

Data = FFh?

Compare the output data at address ADDR to FFh. If the data equals FFh, then that address has been erased. Continue verification until the last address has been verified or until the maximum erase pulse count (1000) has been reached. Typically, most devices will fully erase within 50–100 erase loops.

Last Address → Increment Address

Check the ADDR variable to see if the last address has been verified. If not, increment the ADDR variable and re-write the erase verify command. Remember to write the erase verify command to address ADDR, since the verify command latches the address. Also, if your system has 64K byte segment boundaries, be sure to increment the base pointer every 64K byte addresses.

Write Read Command

After full chip verification, write the read command (00h) to switch the device to read mode. If you plan to reprogram the device immediately, this step is not necessary.

Apply V_{PPL} (Optional)

Switch V_{PP} off. With V_{PP} left on, the command register offers data protection by requiring a precise sequence to initiate programming or erasure. However, V_{PP} controls overall command register operation. Turning V_{PP} off disables the command register, thus providing absolute data protection. Without the high voltage, the reprogramming mechanisms cannot occur and the component becomes a read only memory (ROM).

Abort/Reset

Whenever a system error condition occurs (reset or reboot), write the Reset command (FFh) to each flash memory twice in succession. This is a good initialization practice in systems leaving V_{PP} at 12V. The processor would be unaware if prior to the reset, it had been in the middle of erasure, and this sequence aborts erasure.

The Program Algorithm Illuminated

The full algorithm will not be interpreted here, although a few items should be noted. You can find a conceptual version of the Quick-Pulse Programming algorithm in Chapter 2, Figure 1, and the complete flow chart in Appendix C.

First, similar to erasure, a two-write sequence starts programming. The first write is the Program Set-Up Command which primes the chip for programming. The chip then latches the address and data to be programmed on the second write. You can abort programming by writing the Reset Command twice in succession instead of the data to be programmed.

Second, the device continues programming until commanded to stop by the Program Verify Command. Similar to the Erase Verify Command, this command performs a couple of functions. Internally, it halts programming and latches the given address for verification. Additionally, the command changes the voltages on the memory array's drains, gates, and sources to the program verify configuration.

The cell programming mechanism is self-limiting. However, do not assume that programming all twenty-five 10 μ s operations in one pass is the best way to attain reliable operation. To a certain degree, programming stresses the memory cell. The stress is considerably lower than that applied to EEPROM (2 MV/cm lower to be specific). But why stress a component without cause? The adaptive Quick-Pulse Programming algorithm with its fast program operations, minimizes all stresses and affords the greatest reliability.

Finally, after writing the Program Verify Command to the device, wait a minimum of 6 μ s before reading the device. The time out accounts for the internal slew rate of switching the memory array from the program to program verify configuration. Do not attempt to read from the device before 6 μ s has passed; the device may appear unprogrammed. Your code will then interpret this as a need for extra program operations. It may needlessly reach the 25 operation limit, even though the byte most probably programmed on the first pass.

Ramifications of the Golden Rule

Always follow the erase command with an erase verify command to stop erasure. Interrupt-driven systems must give high priority to servicing the reprogramming timer interrupts. Systems that reset upon a watchdog time-out must reset the flash memory device before rebooting. (See discussion on device initialization.) Likewise any non-maskable interrupt should software- or hardware-reset the flash memory before performing a context switch.

Use an oscilloscope to calibrate all time delays before attempting erasure. The delay modules include the 6 μ s, 10 μ s, and 10 ms timer routines.

Blanket program all addresses to 00h data before erasing. Verify correct implementation of the programming algorithm with a PROM programmer before attempting erasure. (Chapter 4 explains how this can be done.)

16- and 32-bit systems require special attention. Each flash device has its own erase characteristic. Do not assume that if the low byte of a data word is not erased, then the high byte must not be either.

Always follow the listed guidelines and take care while developing your code to eliminate the erase control issue. Consider it similar to implementing any control function. Once the code is debugged and stable, the issue goes away.

You might ask, is it not possible to control erasure through hardware? The alternative to software control is integrated hardware control or an external controller. Intel offers a complete line of high density, cost-effective products with integrated hardware control, often called automation. Whether software or hardware controlled, Intel's ETOX Flash Memory offers the most reliable, dense, manufacturable, and fastest read/write nonvolatile memory. Other EEPROM approaches have drawbacks of multiple transistors per memory cell. This property negatively affects all those attributes offered by Intel's ETOX flash memory.

DEBUGGING YOUR CODE AND OTHER TIPS ON TESTING

As with any software checkout, a few simple principles enable complete flash memory algorithm debug. The following sections offer some hints to make your job easier.

Software Drivers Save You Time

Intel saves you time by offering various processor-family flash memory drivers. You simply edit the files to suit your system. Then assemble the driver, link, and locate it, and you are ready for debug.

These drivers offer the framework for successful flash memory reprogramming, and require some customization to fit your particular system. If your processor's driver is not available, you may use the available driver software as an example. One caution in advance: The drivers have been written in assembly language to give the most speed- and memory- efficient code. However, most people prefer high-level programming.

High-level programming can be used for everything except software-timer generation. Compilers may give different routines with different object code on each compilation. Therefore, the timers must be either hardware-based or coded in assembly language. Software

timers also present some risk if there is a frequency upgrade change on the controlling processor. Regenerate and check your timer routines whenever the system clock rate changes.

Timers, Test Loops and Assembly Level Programming

Timing circuits or software play the most crucial role in flash memory reprogramming. Good timers precisely control their function; sloppy timers produce faults. An example of a sloppy timer is one produced by a compiler. Each time high-level languages recompile, the low-level object coding may change. Thus, a timing loop may be 10 ms one compilation, and much longer or shorter the next time.

You can check your timing method with the following simple technique. Develop test loops which call the various timers' routines. For example, implement the 6 μ s, 10 μ s, and 10 ms timers used with the 28F512 and 28F010. If you have a spare port or peripheral output, use it to trigger an oscilloscope. Follow the trigger call with the timer routines. If you do not have a spare port, write to the flash memory address space before and after each timer call. You can trigger the oscilloscope off of the flash memory CE# signal. Remember to power-down the system and remove the flash memory before attempting these methods.

Once the timer code has been verified, you can link and locate it to a higher-level erase/program algorithm implementation.

Programming—The Key to Proper Erasure

Earlier sections described the importance of programming 00's prior to erasure. This procedure equalizes the charge on all memory cells; following this step all bits erase in unison.

A conclusive debug technique can check your programming software. Simply use your software to program zeroes into the flash memory and then verify this step using a conventional PROM programmer. Load the PROM programmer's buffer with all zeroes and compare the buffer to the flash memory. If your programmer does not service the flash memory, call the company for the latest software upgrade. Alternatively, one can easily rig the 512K flash memory to look like a 512K EPROM. Simply jumper V_{CC} on a 32-pin socket

to a few pins. Note that the 27512 EPROM and 28F512 flash memory have different intelligent identifier codes. Override the identifier code check to use this method. See Figure 7 for socket details.

Some microcontrollers have limited address space or internal memory that masks certain external address space. Even if you do not use sections of the flash memory, you must still access these sections to program zeroes before erasure. Map and decode port bits to access unused address space, and verify that all bytes are programmed to zero before proceeding with erasure.

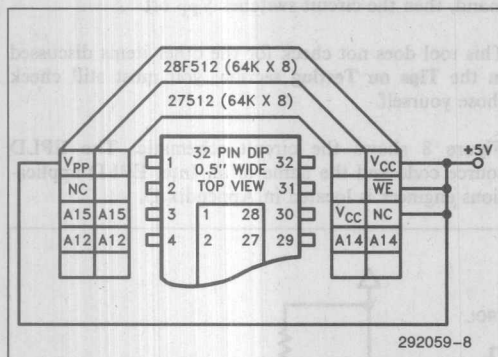


Figure 7. A 28F512 can be read in a PROM programmer as a 27512 by jumpering the appropriate pins to V_{CC} . The same method applies to the 28F010.

16- and 32-Bit Systems—Achieving Optimum Reprogramming Throughput

Erasing flash memory in 16- and 32-bit systems requires special consideration. One could implement the program and erase algorithms in a byte-wise fashion, but this is time-consuming. Alternately, one can treat the multiple flash memory as a data word, and gain optimum performance.

The primary consideration with the latter approach is that one device may program or erase faster than the other(s). Subsequent programming or erasure of a slower device compromises the functionality of the faster device by subjecting it to the slow-device timing.

Consider an example of erasure in a 16-bit system. After 10 passes through the erase operations, both devices verify through address 07C3h. Then at address 07C5h, the processor reads data word 83FFh.

Since erase data is FFFFh, a few bits in the upper byte/device have not erased. The natural flow of the algorithm would dictate another erase operation. But what about the lower device? Could it be completely erased?

Of course it could be; every device erases at a different rate and the algorithm has only checked up to address 07C5h.

You can take advantage of the data rate of wider data buses by utilizing the command register, the reset command and an analysis of erasure. Each erase operation is 10 ms. Each byte verification takes 6 μ s. Therefore, erasure takes three orders of magnitude longer than verification. Optimization for erasure yields the optimum performance because verification is a second order effect.

Let us reconsider the previous example. At address 07C5h, the data word does not verify. On the next erase operation edit the erase (and erase verify) command word such that only the high byte gets the erase command, and the low byte gets a reset command. (i.e., change the command word from 2020h to 20FFh).

See Application Note AP-316 for a detailed flow chart for this approach. Note this document is based on the 28F256; however, most concepts carry through to the higher density devices. (Literature Order number 292046).

Logic Analyzers and In-Circuit Emulators

Many programmers use logic analyzers and in-circuit emulators to debug code. These approaches are fine for flash memory algorithm debug if certain conditions are met.

1. Check timing routines with an oscilloscope; there is no alternative.
2. Know your code and set breakpoints intelligently. One designer had the bad luck of throwing in a breakpoint in the middle of the program 00's loop. After stepping through a couple of byte program 00 loops, he called the erase flow routine. Can you identify the problem?
3. Single step through your reprogramming code, if and only if, the flash memory device is removed from the system.

Testing Your Software—One More Time

Some flash memories specify 100 erase/program cycles. This is a minimum specification; Intel Flash Memories cycle 100,000 times. With this in mind, feel free to check and recheck reprogramming operations. There is no reliability risk in doing so.

One confidence-raising test is similar to that done on systems: stress the system/software by executing test code numerous times consecutively. Set the reprogramming drivers in a loop, and let them run 20–40 times. On each consecutive pass, use a constant data pattern such as 0AAh. This tests the reprogramming code from a quasi-static perspective. Missing is the true system environment. In the true system environment, multiple inputs compete with the flash memory for interrupt priority. Also, RF noise from motors can cause spikes and glitching on V_{PP} or V_{CC} . Additionally, fully loaded systems or partially loaded systems might have different V_{PP} response characteristics or noise levels. Signals that look clean in the lab, might not be all that clean in the true operating environment. Therefore, flash reprogramming tests should be done in the true system environment as a final test.

Watchdog Timer Debug Circuit

This section describes a simple tool you can build for debugging your code. Since Intel's first generation Flash Memories contain integrated stop-timers, this tool offers no real benefit. It is simply shown in case a designer is debugging code for alternate sources. An EPLD watches the flash memory data bus and control signals for the erase sequence—erase set-up, erase, and erase verify commands. Once the CPU initiates erasure, the debug tool starts a 15 ms timer. Should the timer count down prior to receiving the erase verify command, then the circuit switches V_{PP} off.

This tool does not check for the other items discussed in the **Tips on Testing** section; you must still check those yourself.

Figure 8 shows the circuit schematic. The EPLD source code and the name of an Intel EPLD applications engineer is located in Appendix D.

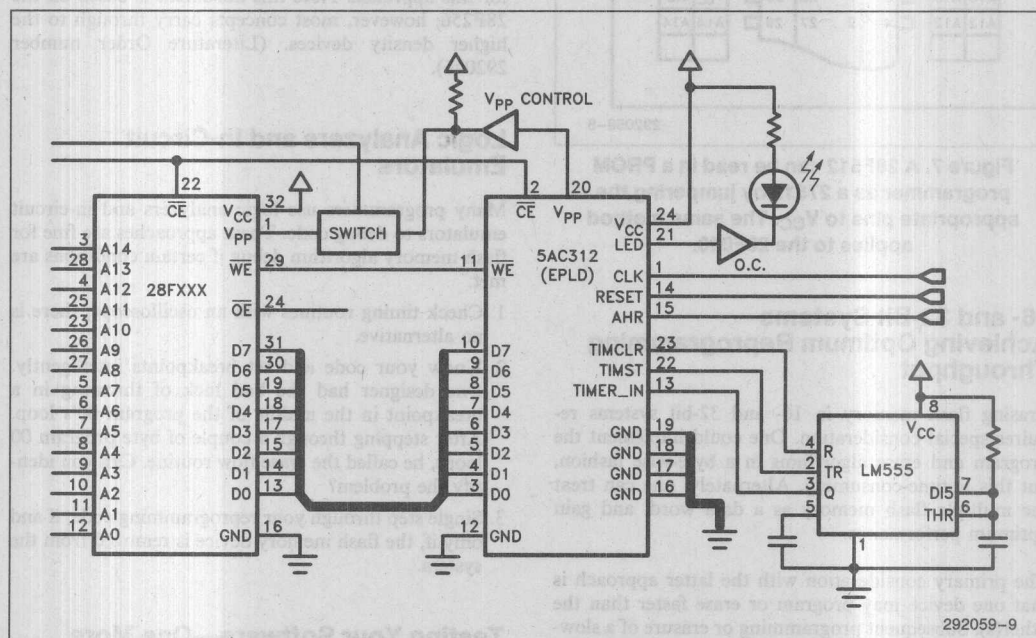


Figure 8. Watchdog Timer Debug Circuit

TROUBLE SHOOTING GUIDE

**Determining the Root Cause—
Software Error vs Device Damage**

The three major indications of a flash memory problem are labeled in the following section. The subsequent paragraphs define potential root causes to investigate.

I. The Device Does Not Program

Did it program before?

A. No.

1. Trigger your oscilloscope on CE# while probing Vpp. Verify that Vpp has reached a steady-state 12V when the device is first written.
2. Set the time-base to 10 μ s/division (the duration of the program operation). Trigger on CE# and probe WE# (look at both traces). Check the duration of the 10 μ s program operation time delay. Also, check the duration of the 6 μ s delay between writing the program verify command and read.
3. Look for ringing on Vpp when Vpp has been switched on. Over-voltage stress on Vpp (ringing with amplitude greater than 13V) will destroy Vpp's silicon structure.
4. Power the system down and back up. Look for destructive glitches on VCC or Vpp (greater than 7V and 13V respectively).
5. Verify erasure and programming on a PROM programmer (if available). Fill the programmer buffer first with 00h data and program the buffer to the flash memory. Then erase the device, and repeat with AAh data. Repeat the last step with 55h data. This sequence fully exercises the array, the input buffers and the output buffers. If all tests pass then check for a hardware or software error.

B. Yes.

1. Have you done anything that may have ESD zapped the devices (i.e., touched the devices while not being grounded, re-wired the proto-board with the components socketed, etc.)? If yes, check part as outlined in section 1.A.5.
2. Have you attempted erasure? If yes, verify your algorithm as outlined in Chapter 4. Also, implement the in-system intelligent identifier mode. If the device outputs an incorrect code, then either an output has been zapped or the golden rule has been violated. Section 1.A.5 describes a method of checking for ESD damage.

II. The Device Does not Erase

Did it erase before?

A. No.

Follow steps 1–5 outlined in section I. When performing step 2, adjust the oscilloscope time base to 10 ms/div.

B. Yes.

Has the board design, clock rate or software changed? System clock rates directly affect the accuracy of software timers. See AP-316 for a discussion on software timing versus clock rate.

III. The Device Erases Spuriously

Exercise all system functions while monitoring the flash memory chip selects. Verify that I/O mapped addresses or logic are not accidentally selecting the flash memory. For example, the space bar character sent from a keyboard controller happens to be 20h. If the flash memory is accidentally selected while this data is on the bus, then erasure will commence on the following cycle when the condition occurs again.

APPENDIX A

TWO APPROACHES TO ALGORITHMS

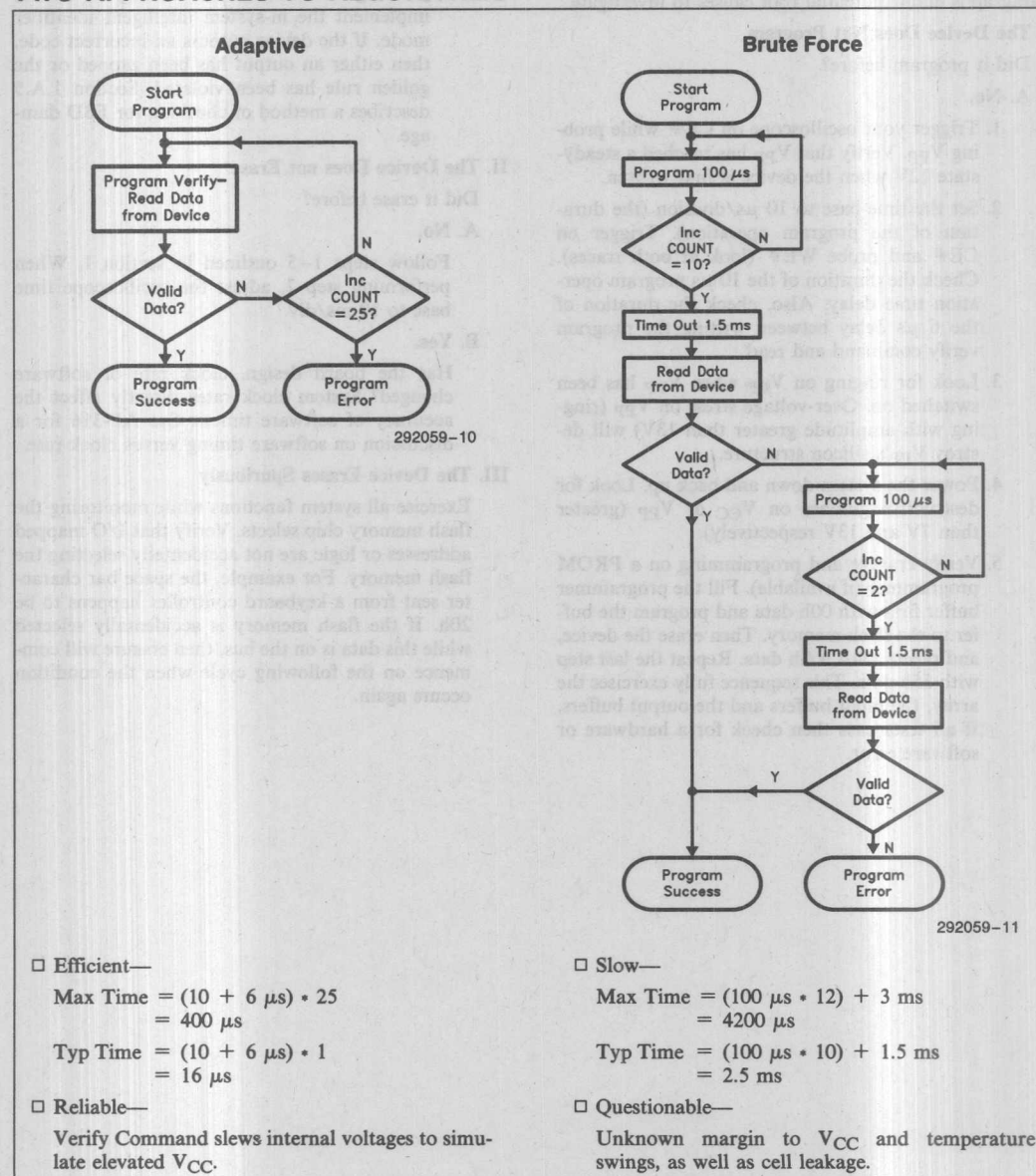


Figure 9. Left and right flow charts compare Intel's (adaptive) Quick-Pulse Programming algorithm and another company's (brute force) approach to flash memory programming.

APPENDIX B

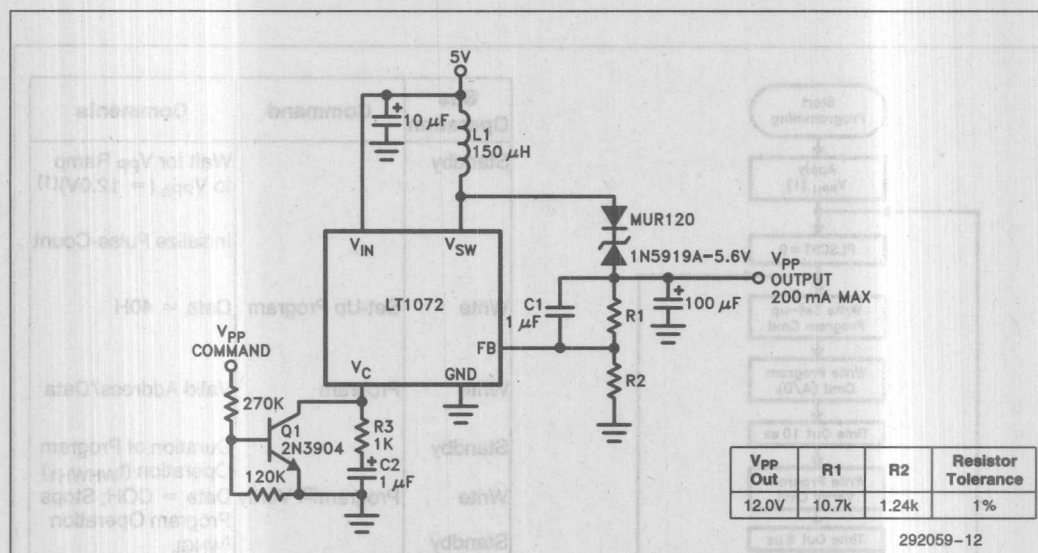
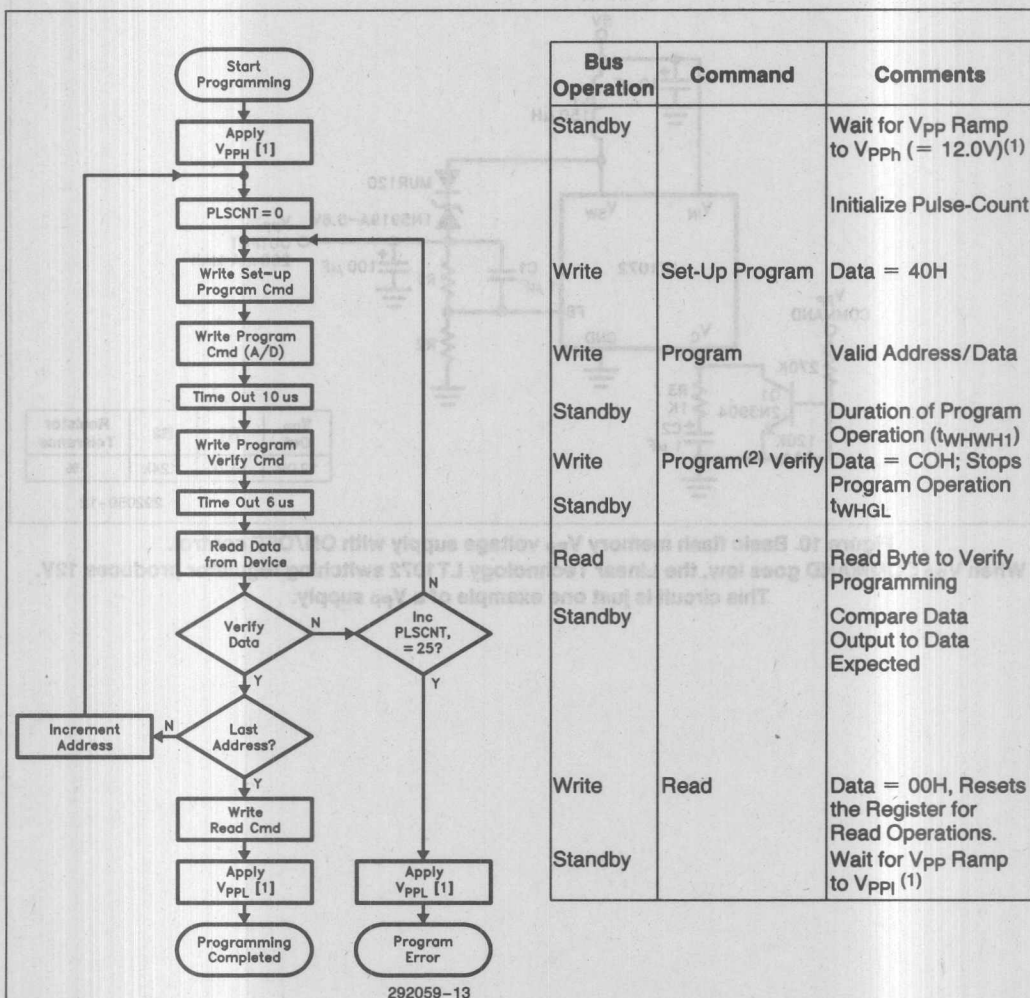


Figure 10. Basic flash memory V_{pp} voltage supply with ON/OFF control.
When V_{pp} COMMAND goes low, the Linear Technology LT1072 switching regulator produces 12V.
This circuit is just one example of a V_{pp} supply.

5

APPENDIX C

QUICK-PULSE PROGRAMMING ALGORITHM

**NOTES:**

1. The V_{pp} power supply can be hard-wired to the device or switchable. When V_{pp} is switched, V_{pp} I may be ground, no-connect with a resistor tied to ground, or less than $V_{CC} + 2.0V$.
2. Program Verify is only performed after byte programming. A final read/compare may be performed (optional) after the register is written with the Read command.
3. CAUTION: The algorithm MUST BE FOLLOWED to ensure proper and reliable operation of the device.

APPENDIX D WATCHDOG TIMER CIRCUIT

EPLD Source Code and Applications Contact Person

Thom Bowns_PLFG Applications

Intel

January 5, 1989

Rev. 008

5AC312

Watchdog timer to cut V_{PP} from FLASH if erase cycle too long.

OPTIONS: TURBO = ON

PART: 5AC312

INPUTS: CLK, D0@3, D1@4, D2@5, D3@6, D4@7, D5@8, D6@9, D7@10,
nCE@2, nWE@11, TIMER_IN@13, RESET@14, AHR@15

OUTPUTS: TIMCLR@23, TIMST@22, LED@21, VPP@20

NETWORK:

```

CLK = INP (CLK)
DO = INP (D0)
D1 = INP (D1)
D2 = INP (D2)
D3 = INP (D3)
D4 = INP (D4)
D5 = INP (D5)
D6 = INP (D6)
D7 = INP (D7)
nCE = INP (nCE)
nWE = INP (nWE)
TIMER_IN = INP (TIMER_IN)
RESET = INP (RESET)
AHR = INP (AHR)           % RESET active high if AHR is high %
COND1 = NOCF (C1d)        % Conditions 1-4 are routed %
COND2 = NOCF (C2d)        % through combinatorial feedbacks %
COND3 = NOCF (C3d)        % to reduce product term count. %
COND4 = NOCF (C4d)        %
CLR = NOCF (CLRd)

```

EQUATIONS:

```

CLRd = RESET * AHR + !RESET * !AHR;
TIMEOUT = /TIMER_IN;
C1d = (/nCE * /nWE * 20H); % Write 20 %
C2d = (/nCE * /nWE * a0H); % Write A0 %
C3d = (/nCE * /nWE * /20H); % Write other than 20 %
C4d = (/nCE * /nWE * /A0H); % Write other than A0 %
20H = /D7 * /D6 * /D5 * /D4 * /D3 * /D2 * /D1 * /D0;
A0H = /D7 * /D6 * /D5 * /D4 * /D3 * /D2 * /D1 * /D0;

```

WATCHDOG
CLOCK: CLK

STATES:	[VPP	LED	TIMST	TIMCLR	XSB]
START	[0	0	0	0	0]
S1	[1	0	0	0	0]
S2	[1	0	1	0	0]
S3	[1	0	1	1	0]
S4	[1	0	0	1	0]
S5	[1	0	1	1	1]
S6	[1	0	1	0	1]
S7	[0	1	1	0	0]

% TRANSITION STATEMENTS %

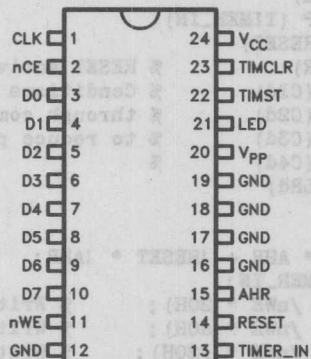
```

START: S1                                % From power up, go to S1 right away %
S1:   IF COND1 THEN S2                   % If write 20, go to next state %
S2:   IF /COND1 THEN S3                  % Until not write 20, hold %
      IF CLR THEN S1
S3:   IF COND1 THEN S4                   % If another write 20, start timer %
      IF COND3 THEN S7                   % If write other than 20, error %
      IF CLR THEN S1
S4:   S5                                % Trigger timer then go to S5 loop %
S5:   IF COND2 THEN S6                   % If write A0, stop timer %
      IF TIMEOUT THEN S7                 % If timer times out, %
                                          go to error state %
      IF COND4 THEN S7                   % If write other than A0, error %
S6:   S1                                % Stop timer and go back to S1 %
S7:   IF CLR THEN S1                     % Error state. wait for a RESET. %

```

END\$

EPLD Pinout 5AC312



292059-14

APPENDIX E

Checklist: Most Common Mistakes that May Lead to Excessive Erasure

- not programming all bytes to 00 data prior to erasure;
- not observing the 6 μ s set-up times between programming or erasure and verification;
- attempting to program before V_{pp} is at 12V (Capacitive Load)
- not latching the erase verify address with the erase verify command, or changing the address on the subsequent read cycle;

Chapters three and four discuss the correct methods of developing and debugging code to diminish the possibility of making these mistakes.

**ENGINEERING
REPORT****Intel Flash Memory
28F256A
28F512
28F010
28F020**

October 1993

Intel Flash Memory

28F256A, 28F512, 28F010, 28F020

CONTENTS	PAGE
INTRODUCTION	5-186
TECHNOLOGY OVERVIEW	5-186
DEVICE ARCHITECTURE	5-186

The command port consists of a command register, command decoder and state latch. The command decoder output and the address latch. The command decoder output directs the operation of the high voltage flash array switch, program voltage generator, and the erase/write program verify voltage generator.

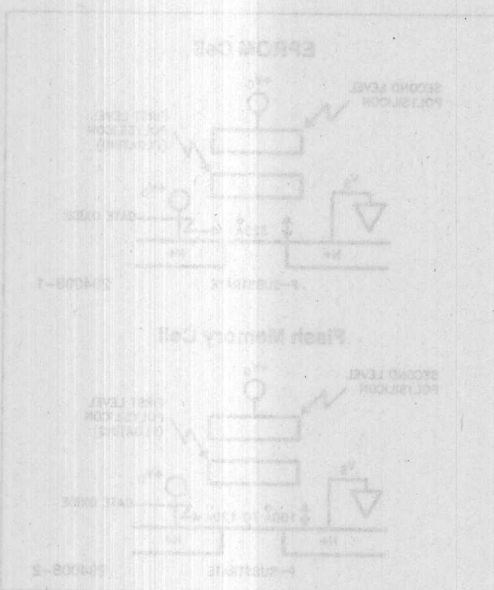


Figure 1. EPROM Cell vs. Flash Memory Cell

The command port consists of a command register, command decoder and state latch. The command decoder output and the address latch. The command decoder output directs the operation of the high voltage flash array switch, program voltage generator, and the erase/write program verify voltage generator.

Functions are selected via the command port in a microprocessor while code contained in the Flash memory is updated on the falling edge of \overline{W} (write enable). The data registers and internal operations.

CONTENTS	PAGE
DEVICE RELIABILITY	5-187
SUMMARY	5-188

The command port interface, internal margin voltage, read timing parameters on Intel's 28F010, 28F020, 28F512 and 28F256A are equivalent to those of CMOS EPROM, EPROM, and SRAM. Access times as fast as 60 ns result from a memory cell center of approximately 20 nA, low resistance polysilicide word lines, advanced access property transistor, and an optimized data-out buffer.

Read timing parameters on Intel's 28F010, 28F020, 28F512 and 28F256A are equivalent to those of CMOS EPROM, EPROM, and SRAM. Access times as fast as 60 ns result from a memory cell center of approximately 20 nA, low resistance polysilicide word lines, advanced access property transistor, and an optimized data-out buffer.

Read timing parameters on Intel's 28F010, 28F020, 28F512 and 28F256A are equivalent to those of CMOS EPROM, EPROM, and SRAM. Access times as fast as 60 ns result from a memory cell center of approximately 20 nA, low resistance polysilicide word lines, advanced access property transistor, and an optimized data-out buffer.

TECHNOLOGY OVERVIEW

Intel's EPROM flash memory technology is derived from its standard CMOS EPROM process base. Using advanced 1.0 μ m and 0.8 μ m double-poly silicon n-well CMOS technology, Intel Flash Memory employs a 1.8 μ m x 4.0 μ m single transistor cell (1.0 μ m) technology and a 3.2 μ m x 2.0 μ m single transistor cell (0.8 μ m technology), affording equivalent array density as comparable EPROM technology. The flash memory cell structure is identical to the EPROM structure, except for the access gate (tunnel oxide). Figure 1 compares the flash memory cell to the EPROM cell.

Flash memory is organized under the single floating gate architecture in a 1T1R1C1 structure. All cells in the array are simultaneously erased via Fowler-Nordheim tunneling. Applying 12V on the source junctions and grounding the select gates erases the entire array in one second (typical). Programming is accomplished with the standard EPROM mechanism of hot electron injection from the cell drain junction to the floating gate. Programming is initiated by bringing both the select gate and the cell drain to high voltage. Programming occurs at a rate of 10 ns pulses per byte.

INTRODUCTION

Intel's ETOX (EPROM tunnel oxide) Flash Memory adds electrical chip erasure and reprogramming to EPROM non-volatility and ease of use. Advances in tunnel oxides and photolithography have made it possible to develop a double-polysilicon single-transistor read/write random access nonvolatile memory, capable of greater than 100,000 reprogramming cycles. Intel Flash Memory electrically erases all bits in the array matrix via electron tunneling. The EPROM programming mechanism of hot electron injection is employed for electrical byte programming.

A command port interface, internal margin voltage generation, power up/down protection and address and data latches augment standard EPROM circuitry to optimize Intel's Flash Memory for microprocessor-controlled reprogramming.

Read timing parameters on Intel's 28F256A, 28F512, 28F010 and 28F020 are equivalent to those of CMOS EPROMs, EEPROMs, and SRAMs. Access times as fast as 65 ns result from a memory cell current of approximately 50 μ A, low resistance poly-silicide word-lines, advanced scaled periphery transistors, and an optimized data-out buffer.

TECHNOLOGY OVERVIEW

Intel's ETOX flash memory technology is derived from its standard CMOS EPROM process base. Using advanced 1.0 μ m and 0.8 μ m double-polysilicon n-well CMOS technology, Intel Flash Memory employs a 3.8 μ m x 4.0 μ m single transistor cell (1.0 μ m technology) and a 2.5 μ m x 2.9 μ m single transistor cell (0.8 μ m technology), affording equivalent array density as comparable EPROM technology. The flash memory cell structure is identical to the EPROM structure, except for the thinner gate (tunnel) oxide. Figure 1 compares the flash memory cell to the EPROM cell.

High quality tunnel oxide under the single floating polysilicon gate promotes electrical erasure. All cells in the array are simultaneously erased via Fowler-Nordheim tunneling. Applying 12V on the source junctions and grounding the select gates erases the entire array in one second (typical). Programming is accomplished with the standard EPROM mechanism of hot electron injection from the cell drain junction to the floating gate. Programming is initiated by bringing both the select gate and the cell drain to high voltage. Programming occurs at a rate of 10 μ s pulses per byte.

DEVICE ARCHITECTURE

Command Port

One feature which differentiates Intel's Flash Memory is the command port architecture, illustrated in Figure 2.

The command port simplifies microprocessor control of the erase, erase verify, program, program verify, and read operations, without the need for additional control pins or the multiplexing of high voltage with control functions. On-chip address and data latches minimize system interface logic and free the system bus during erase and program operations. High voltage (12V) on the Vpp pin enables the command port. In the absence of this high voltage, the command port defaults to the read operation, inhibiting erasure or programming of the device.

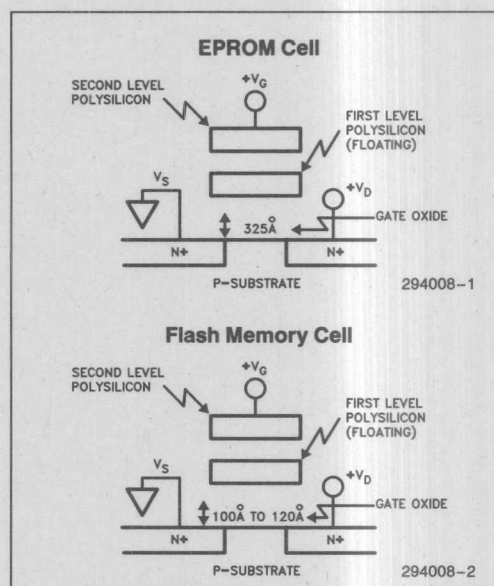


Figure 1. EPROM Cell vs. Flash Memory Cell

The command port consists of a command register, command decoder and state latch, the data-in latch, and the address latch. The command decoder output directs the operation of the high voltage flash-erase switch, program voltage generator, and the erase/program verify voltage generator.

Functions are selected via the command port in a microprocessor write cycle controlled by the Chip-Enable and Write-Enable pins. Contents of the address latch are updated on the falling edge of Write-Enable. The rising edge of Write-Enable latches the command and data registers, and initiates operations.

Erase

Erase is achieved through a two-step write sequence. The erase set-up code is written to the command register in the first cycle. The erase confirmation code is written in the second cycle. The rising edge of this second Write-Enable pulse initiates the erase operation. The command decoder triggers the high voltage flash-erase switch, connecting the 12V supply to the source of all bits in the array, while all wordlines are grounded. Fowler-Nordheim tunneling results in the simultaneous erasure of all bits.

The array source switch, shown in Figure 3, switches high voltage onto the source junctions. During erasure, the high voltage latch formed by M_5 through M_8 enables transistor M_{15} . Transistor M_{15} pulls the array source up to 12V. Transistor M_{16} pulls the source to ground during read and program operations.

To obtain fast erase times, the device must supply the grounded gate breakdown current which occurs on the sources of the memory array. The upper boundary for current sourcing capability of M_{15} is set by the maximum allowable substrate current. If V_{pp} is raised to 12V before V_{CC} is above approximately 1.8V, the low V_{CC} detect circuit formed by transistors M_1 to M_4 drives the node LOW V_{CC} to 9V. Transistors M_9 to M_{11} then force the erase circuit into a non-erase state with M_{15} off and M_{16} on. When V_{CC} rises above 1.8V, the chip will be reset into the read state.

Writing the erase verify code into the command register terminates erasure, latches the address of the byte to verify, and sets the internally-generated erase margin voltage. The microprocessor then accesses the output from the addressed byte using standard read timings. The verify procedure repeats for all addresses. Should a byte require more time to reach the erased state, another erase operation is applied. The erase and verify operations continue until the entire array is erased.

Programming

Programming follows a similar flow. The program set-up command is written to the command register on the first cycle. The second cycle loads the address and data latches. The rising edge of the second Write-Enable pulse initiates programming by applying high voltage to the gates and drains of the bits to be programmed.

Writing the program verify command to the register terminates the programming operation and applies the program verify voltage to the newly programmed byte. Again, the addressed byte can be read using standard microprocessor read timings. Should the addressed byte require more time to reach the programmed state, the programming operation and verification are repeated until the byte is programmed.

DEVICE RELIABILITY

Cell Margining

Erase and program verification ensure the data retention of the newly altered memory bits. The cell margining performed in the Quick-Pulse Programming and Quick-Erase algorithms is more reliable than historical overpulsing schemes as margining tests the amount of charge stored on the floating gate.

Intel's 28F256A through 28F020 Flash Memories employ a unique circuit to internally generate the erase and program verify voltages. Figure 4 shows a simplified version of the circuit. The circuit consists of a high voltage switch and the verify voltage generator. Transistors M_1 through M_4 constitute the high voltage switch which disconnects V_{pp} from the resistor when the device is not in the verify mode. The verify voltage generator includes a resistor divider and a buffer. Internal margin voltage generation maintains microprocessor compatibility by eliminating the need for external reference voltages.

Erase/Program Cycling

One of the most significant aspects of Intel Flash Memory is its capability for greater than 100,000 erase/program cycles. Destructive oxide breakdown has been a limiting factor in extended cycling of thin oxide EEPROMs. Intel's ETOX flash memory technology extends cycling performance through: improved tunnel oxide processing that increases charge carrying capability ten-fold; reduced oxide area under stress minimizing probability of oxide defects in the region; and reduced oxide stress due to a lower peak electric field (lower erase voltage than EEPROM).

A typical cell erase/program margin (V_t) is shown as a function of reprogramming cycles in Figure 5. After 100,000 reprogramming cycles, a 2.5V program read margin exists, ensuring reliable data retention.

Reliable erase/program cycling also requires proper selection of the erase V_t maximum and maintenance of a tight V_t distribution. The maximum erased V_t is set to 3.2V via the erase algorithm and the internal erase verify circuits. Superior oxide quality gives an erased V_t distribution width that improves slightly with cycling (Figure 6). The tight erase V_t distribution gives an order of magnitude of erase time margin to the fastest erasing cell (Figure 7).

SUMMARY

Intel's ETOX flash memory technology is a breakthrough in adding electrical chip-erase to high-density EPROM technology. Intel's CMOS Flash Memory offers the most cost-effective and reliable alternative for read/write random access non-volatile memory. Microprocessor-compatible specifications, straightforward interfacing, and in circuit alterability allow designers to easily augment memory flexibility and satisfy the need for nonvolatile storage in today's designs.

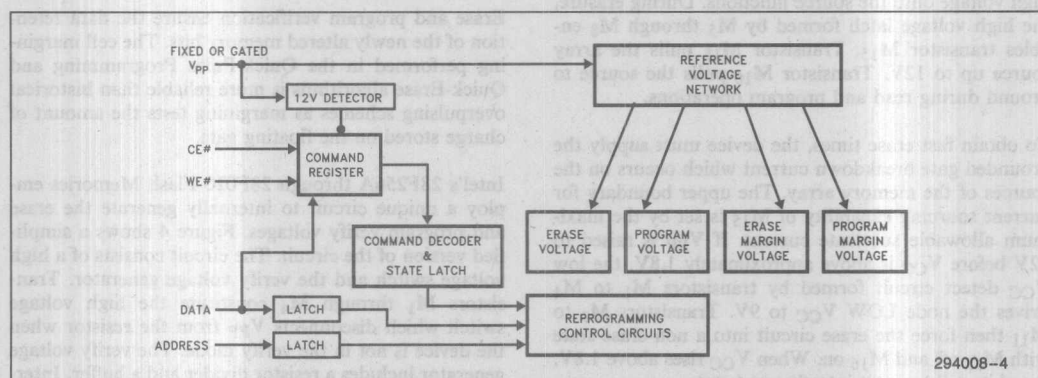


Figure 2. Command Port Block Diagram

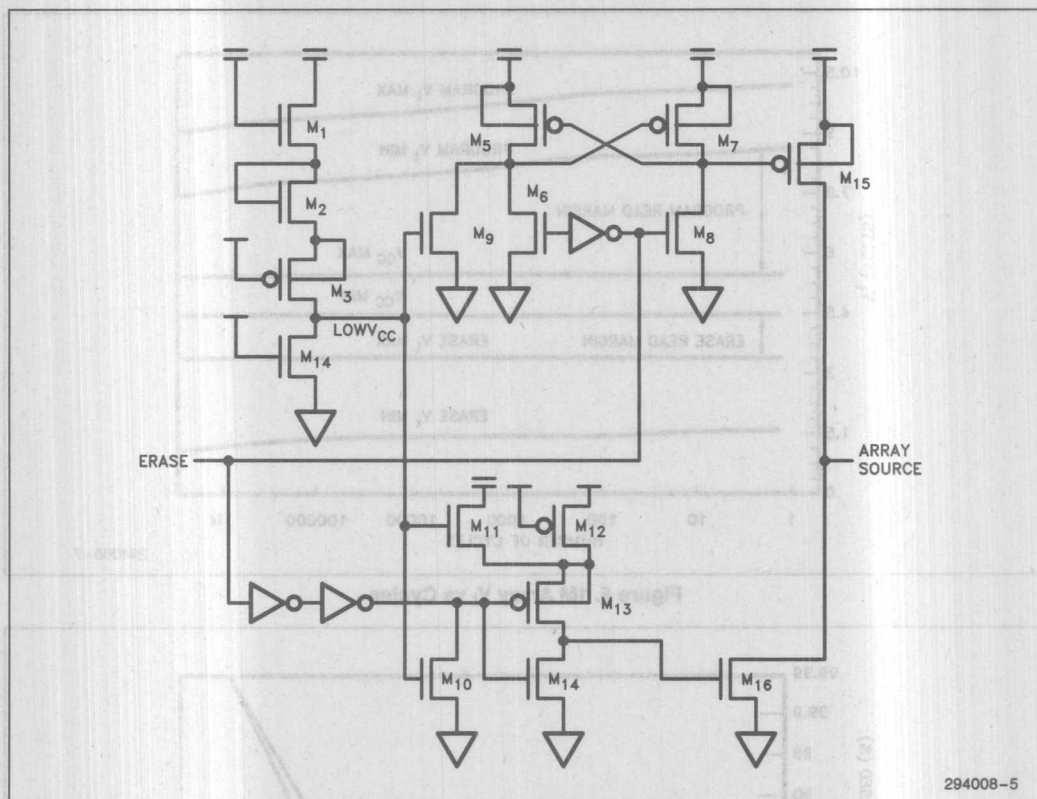


Figure 3. Array Source Switch

5

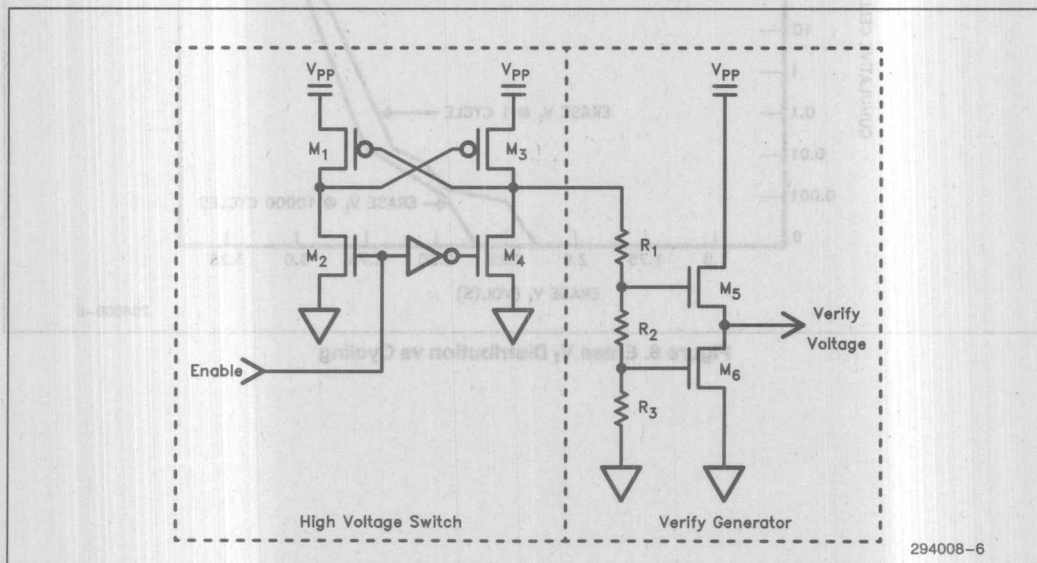
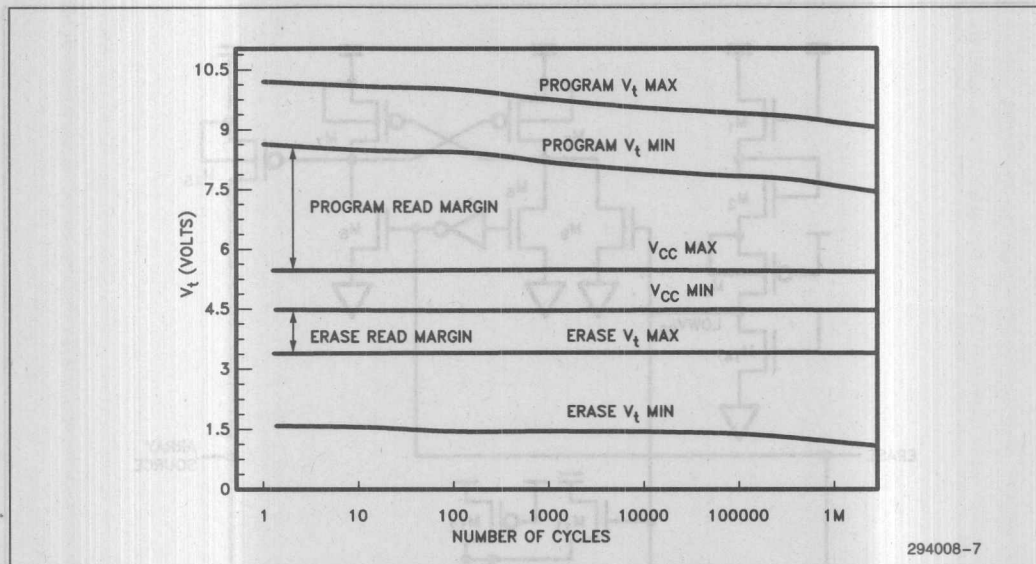
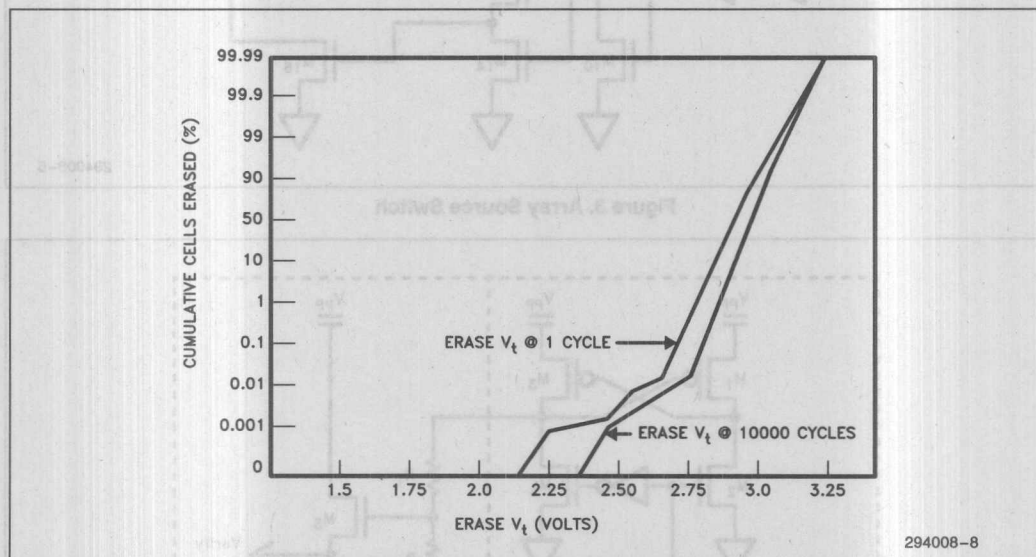
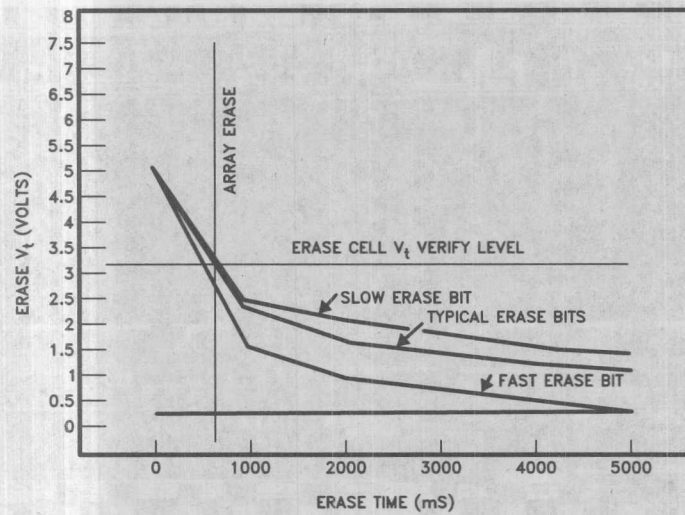


Figure 4. Erase/Program Verify Generator

Figure 5. 1M Array V_t vs CyclesFigure 6. Erase V_t Distribution vs Cycling



294008-9

Figure 7. Array Erase V_t vs Erase Time

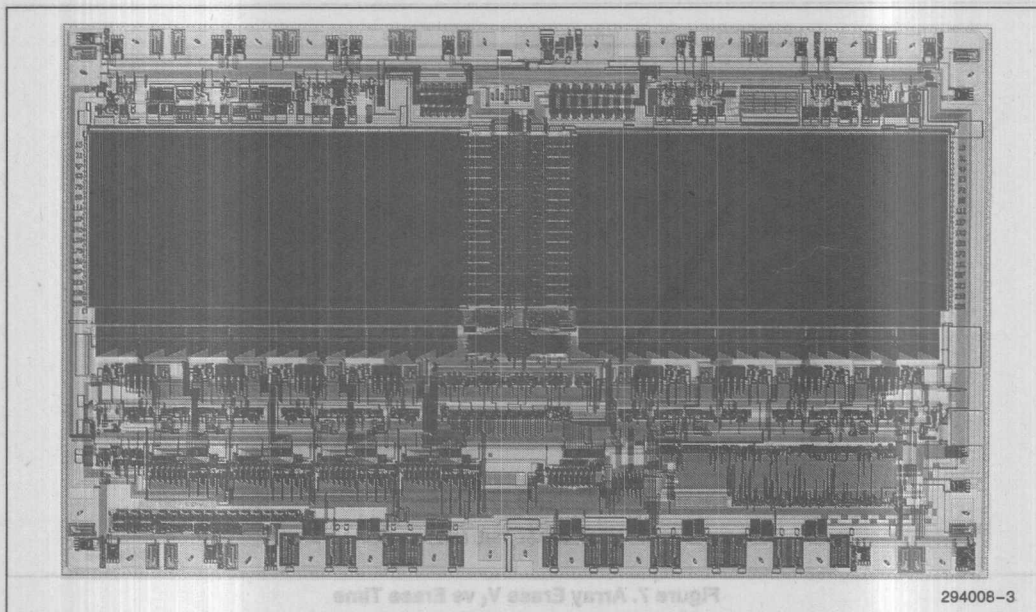
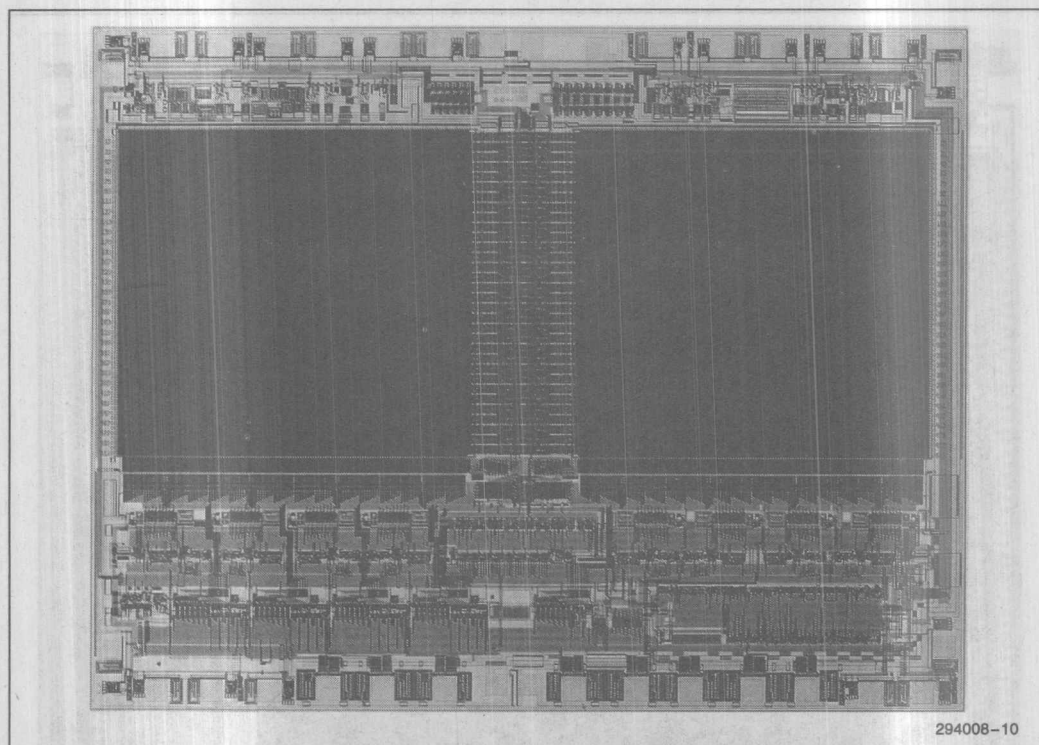


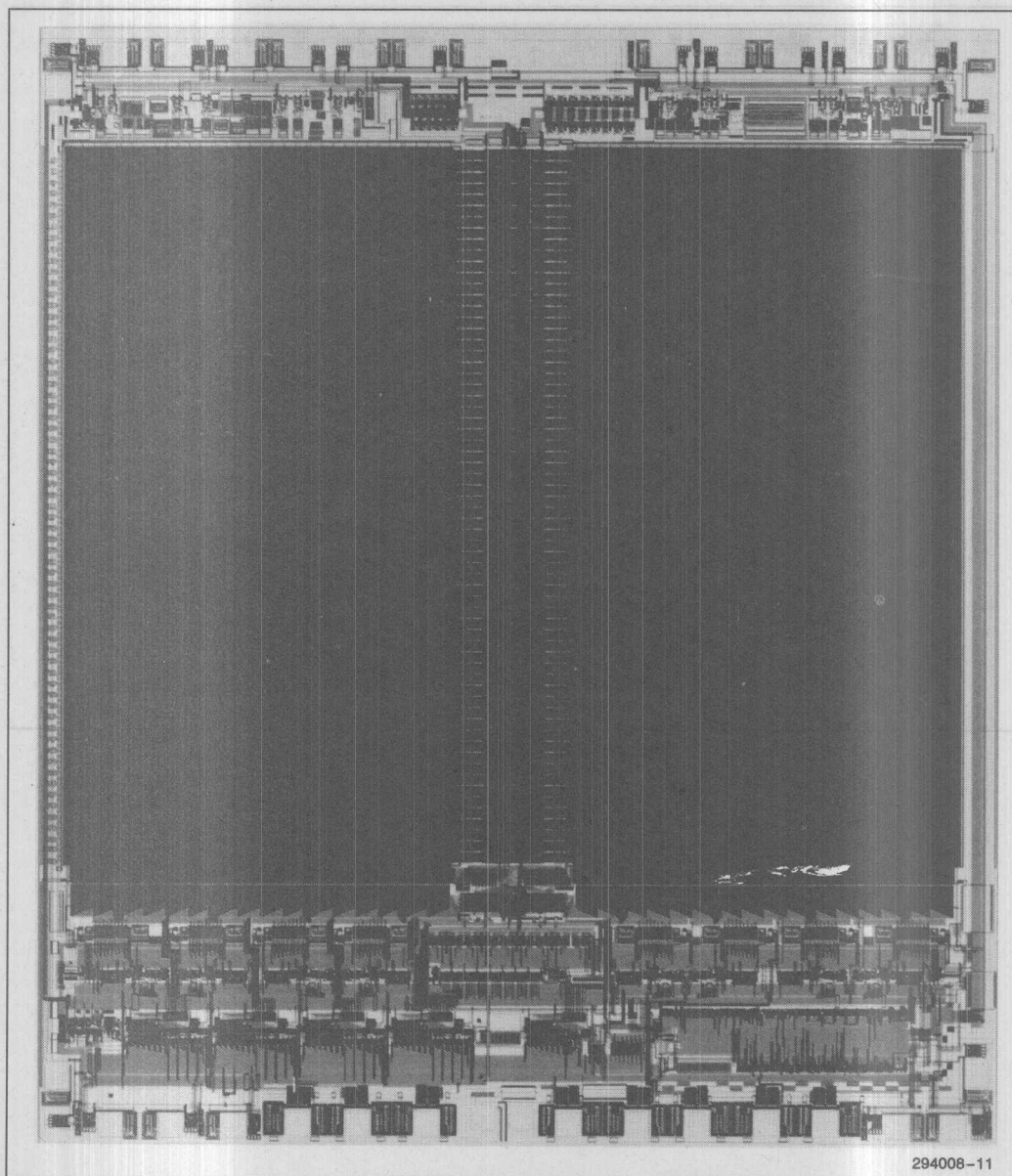
Figure 8. 28F256A Die Photograph



294008-10

Figure 9. 28F512 Die Photograph

5



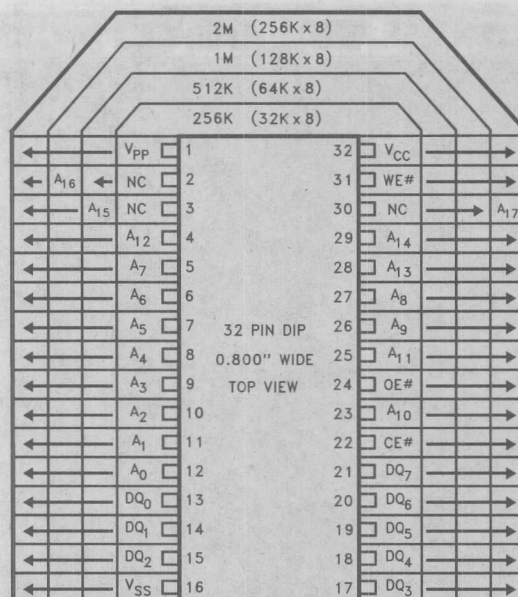
294008-11

Figure 10. 28F010 (1.0μ) Die Photograph



294008-12

Figure 11. 28F020 Die (1.0μ) Photograph



294008-17

Figure 12. Flash Memory Pinouts

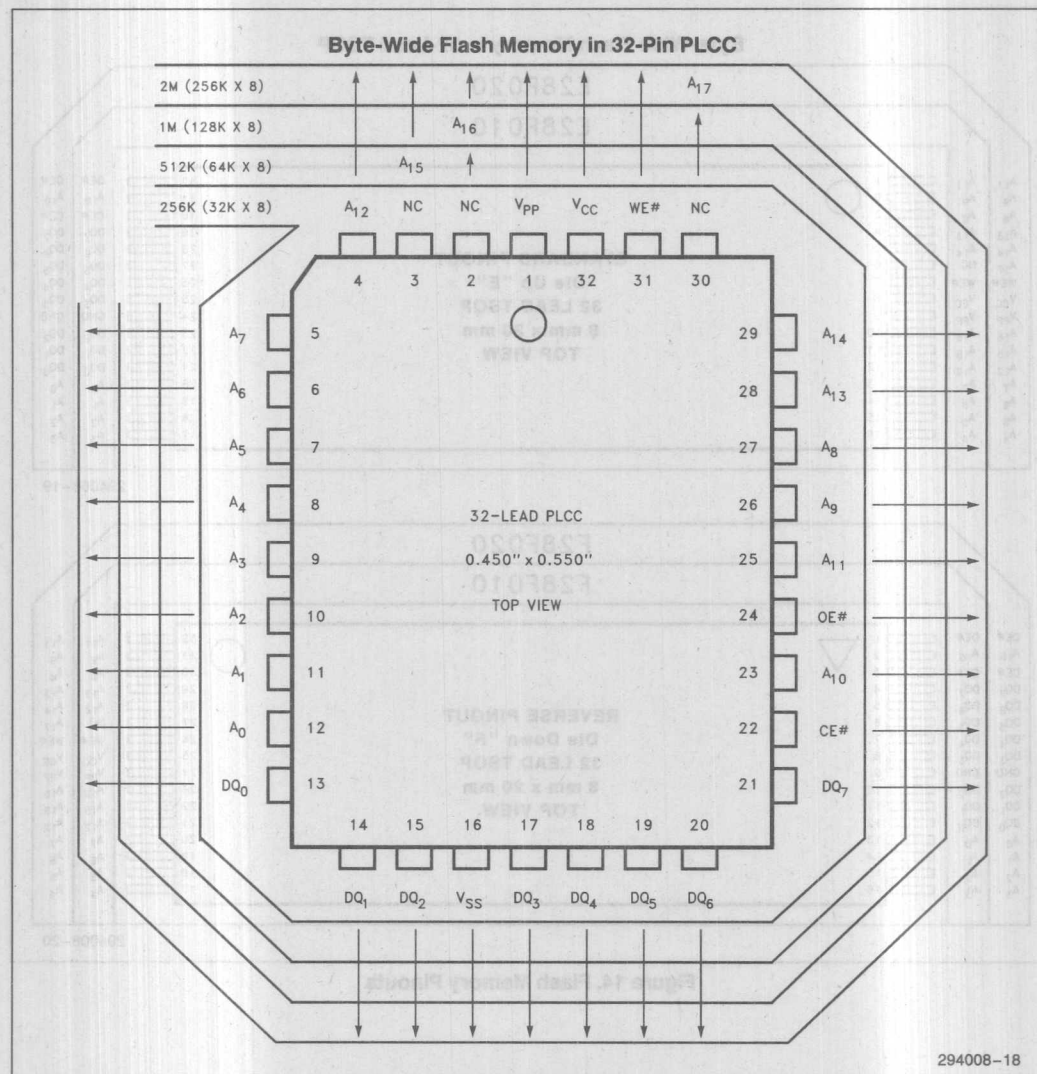


Figure 13. Flash Memory Pinouts

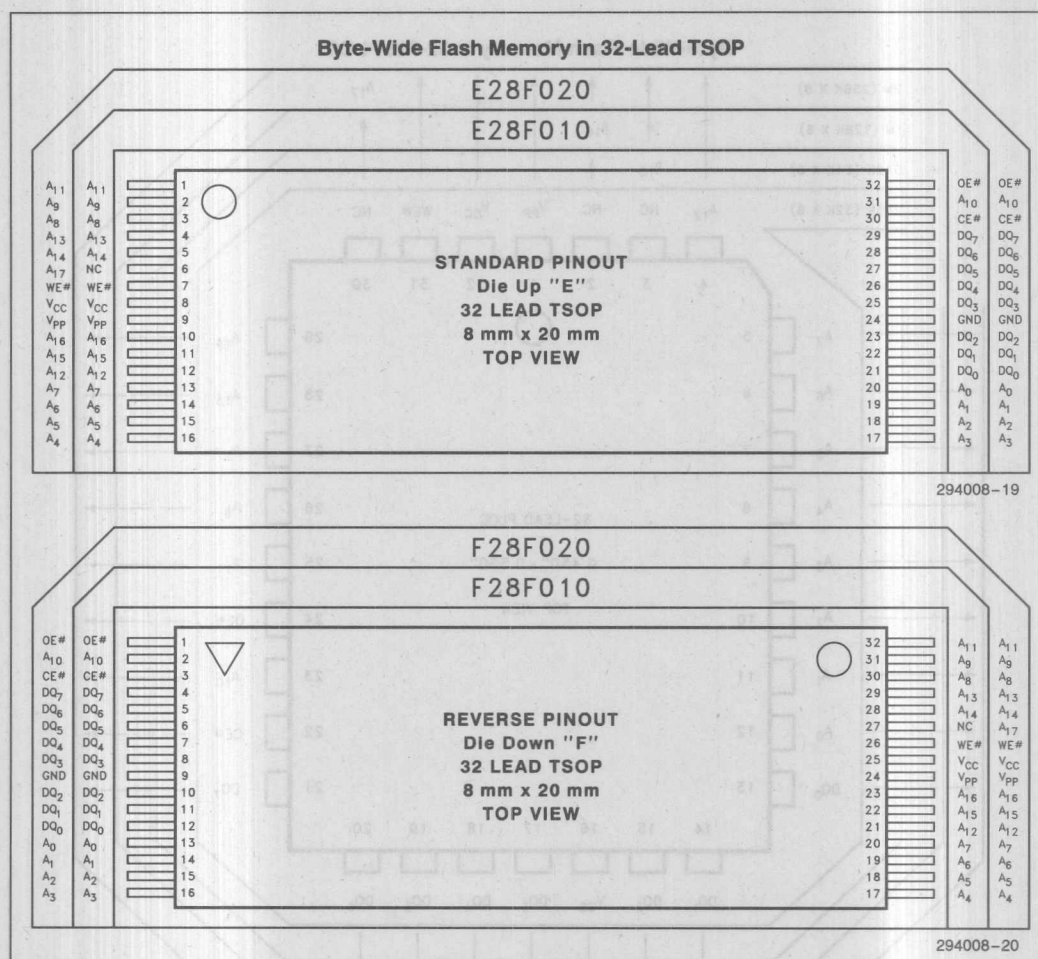


Figure 14. Flash Memory Pinouts

Columns are number 0 through 511 beginning with the column nearest the X-decoder.
Outputs are grouped as follows:

Array Organization:							Left Half Array IO ₀ IO ₁ IO ₂ IO ₃ BL ₃₈₄ ← BL ₀		Right Half Array IO ₄ IO ₅ IO ₆ IO ₇ BL ₀ → BL ₃₈₄		
Address							Bitlines				
A ₁₄	A ₁₂	A ₁₀	A ₂	A ₁	A ₀	A ₃	IO ₀ & IO ₇	IO ₁ & IO ₆	IO ₂ & IO ₅	IO ₃ & IO ₄	
0	0	0	0	0	0	0	BL ₃₈₄	BL ₂₅₆	BL ₁₂₈	BL ₀	
0	0	0	0	0	0	1	BL ₃₈₅	BL ₂₅₇	BL ₁₂₉	BL ₁	
0	0	0	0	0	0	1	0	BL ₃₈₆	BL ₂₅₈	BL ₁₃₀	BL ₂
0	0	0	0	0	0	1	1	BL ₃₈₇	BL ₂₅₉	BL ₁₃₁	BL ₃
0	0	0	0	1	0	0		BL ₃₈₈	BL ₂₆₀	BL ₁₃₂	BL ₄
0	0	0	0	1	0	1		BL ₃₈₉	BL ₂₆₁	BL ₁₃₃	BL ₅
0	0	0	0	1	1	0		BL ₃₉₀	BL ₂₆₂	BL ₁₃₄	BL ₆
0	0	0	0	1	1	1		BL ₃₉₁	BL ₂₆₃	BL ₁₃₅	BL ₇
•	•	•	•	•	•	•	•	•	•	•	•
1	1	1	1	1	0	0		BL ₅₀₈	BL ₃₈₀	BL ₂₅₂	BL ₁₂₄
1	1	1	1	1	0	1		BL ₅₀₉	BL ₃₈₁	BL ₂₅₃	BL ₁₂₅
1	1	1	1	1	1	0		BL ₅₁₀	BL ₃₈₂	BL ₂₅₄	BL ₁₂₆
1	1	1	1	1	1	1		BL ₅₁₁	BL ₃₈₃	BL ₂₅₅	BL ₁₂₇

Figure 15. 28F256A Bitline Decoding

X Address								Row
A ₇	A ₆	A ₅	A ₄	A ₁₃	A ₁₁	A ₉	A ₈	WL
0	0	0	0	0	0	0	0	XL ₀
0	0	0	0	0	0	0	1	XL ₁
0	0	0	0	0	0	1	0	XL ₂
0	0	0	0	0	0	1	1	XL ₃
0	0	0	0	0	1	0	0	XL ₄
0	0	0	0	0	1	0	1	XL ₅
0	0	0	0	0	1	1	0	XL ₆
0	0	0	0	0	1	1	1	XL ₇
0	0	0	0	1	0	0	0	XL ₈
0	0	0	0	1	0	0	1	XL ₉
0	0	0	0	1	0	1	0	XL ₁₀
0	0	0	0	1	0	1	1	XL ₁₁
0	0	0	0	1	1	0	0	XL ₁₂
0	0	0	0	1	1	0	1	XL ₁₃
0	0	0	0	1	1	1	0	XL ₁₄
0	0	0	0	1	1	1	1	XL ₁₅
0	0	0	1	1	1	1	1	XL ₁₆
0	0	0	1	1	1	1	0	XL ₁₇
0	0	0	1	1	1	0	1	XL ₁₈
0	0	0	1	1	0	0	0	XL ₁₉
0	0	0	1	1	0	1	1	XL ₂₀
0	0	0	1	1	0	1	0	XL ₂₁
0	0	0	1	1	0	0	1	XL ₂₂
0	0	0	1	1	0	0	0	XL ₂₃
0	0	0	1	0	1	1	1	XL ₂₄
0	0	0	1	0	1	1	0	XL ₂₅
0	0	0	1	0	1	0	1	XL ₂₆
0	0	0	1	0	1	0	0	XL ₂₇
0	0	0	1	0	0	1	1	XL ₂₈
0	0	0	1	0	0	1	0	XL ₂₉
0	0	0	1	0	0	0	1	XL ₃₀
0	0	0	1	0	0	0	0	XL ₃₁

Figure 16. 28F256A Wordline Decoding

								row
A ₇	A ₆	A ₅	A ₄	A ₁₃	A ₁₁	A ₉	A ₈	WL
0	0	1	0	0	0	0	0	XL32
•	•	•	•	•	•	•	•	•••
0	0	1	0	1	1	1	1	XL47
0	0	1	1	1	1	1	1	XL48
•	•	•	•	•	•	•	•	•••
0	0	1	1	0	0	0	0	XL63
0	1	0	0	0	0	0	0	XL64
•	•	•	•	•	•	•	•	•••
0	1	0	0	1	1	1	1	XL79
0	1	0	1	1	1	1	1	XL80
•	•	•	•	•	•	•	•	•••
0	1	0	1	0	0	0	0	XL95
1	1	1	0	0	0	0	0	XL234
•	•	•	•	•	•	•	•	•••
1	1	1	0	1	1	1	1	XL249
1	1	1	1	1	1	1	1	XL250
•	•	•	•	•	•	•	•	•••
1	1	1	1	0	0	0	0	XL255

Figure 16. 28F256A Wordline Decoding (Continued)

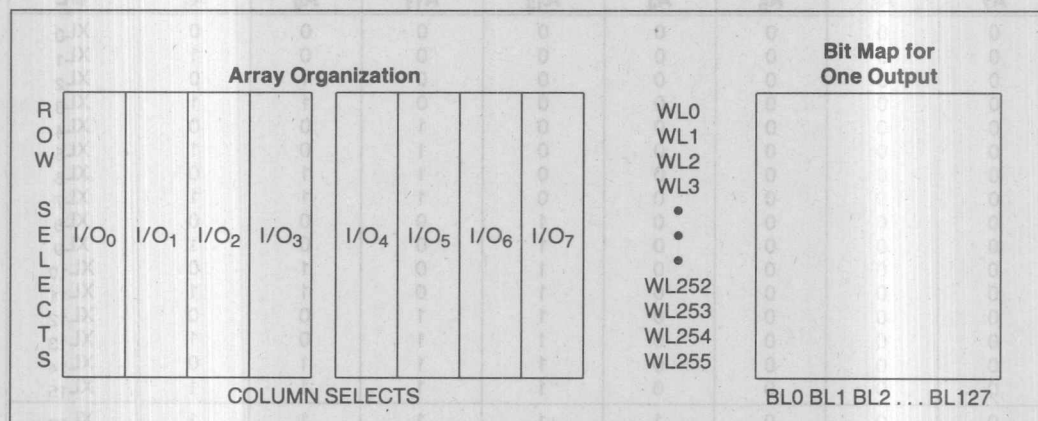


Figure 17. 28F256A Bit Map

Columns are numbered 0 through 511 beginning with the column nearest the X-decoder.

Outputs are grouped as follows:

							Left Half Array				Right Half Array			
							O0	O1	O2	O3	O4	O5	O6	O7
Address							Bitlines							
A14	A15	A3	A10	A2	A1	A0	IO0/7	IO1/06	IO2/05	IO3/04				
0	0	0	0	0	0	0	BL384	BL256	BL128	BL0				
0	0	1	0	0	0	0	BL385	BL257	BL129	BL1				
0	0	0	0	0	0	1	BL386	BL258	BL130	BL2				
0	0	1	0	0	0	1	BL387	BL259	BL131	BL3				
0	0	0	0	0	1	0	BL388	BL260	BL132	BL4				
0	0	1	0	0	1	0	BL389	BL261	BL133	BL5				
0	0	0	0	0	1	1	BL390	BL262	BL134	BL6				
0	0	1	0	0	1	1	BL391	BL263	BL135	BL7				
•	•	•	•	•	•	•	•	•	•	•				
1	1	0	1	1	1	0	BL508	BL380	BL252	BL124				
1	1	1	1	1	1	0	BL509	BL381	BL253	BL125				
1	1	0	1	1	1	1	BL510	BL382	BL254	BL126				
1	1	1	1	1	1	1	BL511	BL383	BL255	BL127				

Figure 18. 28F512 Bitline Decoding

X-DECODING: Wordlines are numbered 0 through 511 beginning at the top of the array.

X Address									Row
A12	A7	A6	A5	A4	A13	A11	A9	A8	WL
0	0	0	0	0	0	0	0	0	XL0
0	0	0	0	0	0	0	0	1	XL1
0	0	0	0	0	0	0	1	0	XL2
0	0	0	0	0	0	0	1	1	XL3
0	0	0	0	0	0	1	0	0	XL4
0	0	0	0	0	0	1	0	1	XL5
0	0	0	0	0	0	1	1	0	XL6
0	0	0	0	0	0	1	1	1	XL7
0	0	0	0	0	1	0	0	0	XL8
0	0	0	0	0	1	0	0	1	XL9
0	0	0	0	0	1	0	1	0	XL10
0	0	0	0	0	1	0	1	1	XL11
0	0	0	0	0	1	1	0	0	XL12
0	0	0	0	0	1	1	0	1	XL13
0	0	0	0	0	1	1	1	0	XL14
0	0	0	0	0	1	1	1	1	XL15
0	0	0	0	1	1	1	1	1	XL16
0	0	0	0	1	1	1	1	0	XL17
0	0	0	0	1	1	1	0	1	XL18
0	0	0	0	1	1	1	0	0	XL19
0	0	0	0	1	1	0	1	1	XL20
0	0	0	0	1	1	0	1	0	XL21
0	0	0	0	1	1	0	0	1	XL22
0	0	0	0	1	1	0	0	0	XL23
0	0	0	0	1	0	1	1	1	XL24
0	0	0	0	1	0	1	1	0	XL25
0	0	0	0	1	0	1	0	1	XL26
0	0	0	0	1	0	1	0	0	XL27
0	0	0	0	1	0	0	1	1	XL28
0	0	0	0	1	0	0	1	0	XL29
0	0	0	0	1	0	0	0	1	XL30
0	0	0	0	1	0	0	0	0	XL31

Figure 19. 28F512 Wordline Decoding

X-DECODING: Wordlines are number 0 through 511 beginning at the top of the array.

X Address									Row
A12	A7	A6	A5	A4	A13	A11	A9	A8	WL
0	0	0	1	0	0	0	0	0	XL32
•	•	•	•	•	•	•	•	•	•
0	0	0	1	0	1	1	1	1	XL47
0	0	0	1	1	1	1	1	1	XL48
•	•	•	•	•	•	•	•	•	•
0	0	0	1	1	0	0	0	0	XL63
0	0	1	0	0	0	0	0	0	XL64
•	•	•	•	•	•	•	•	•	•
0	0	1	0	0	1	1	1	1	XL79
0	0	1	0	1	1	1	1	1	XL80
•	•	•	•	•	•	•	•	•	•
0	0	1	0	1	0	0	0	0	XL95
1	1	1	1	0	0	0	0	0	XL480
•	•	•	•	•	•	•	•	•	•
1	1	1	1	0	1	1	1	1	XL495
1	1	1	1	1	1	1	1	1	XL496
•	•	•	•	•	•	•	•	•	•
1	1	1	1	1	0	0	0	0	XL511

Figure 19. 28F512 Wordline Decoding (Continued)

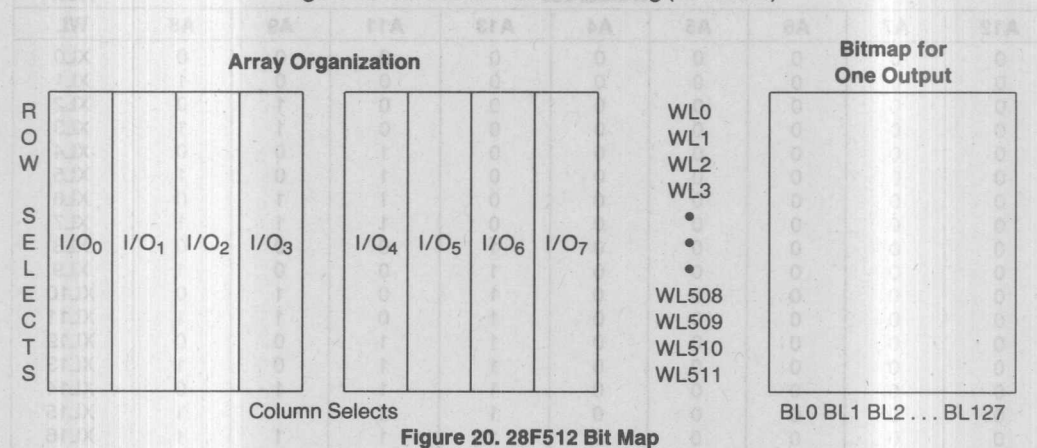


Figure 20. 28F512 Bit Map

Columns are number 0 through 511 beginning with the column nearest the X-decoder.
Outputs are grouped as follows:

Array Organization:							Left Half Array IO ₀ IO ₁ IO ₂ IO ₃ BL ₃₈₄ ← BL ₀		Right Half Array IO ₄ IO ₅ IO ₆ IO ₇ BL ₀ → BL ₃₈₄	
Address							Bitlines			
A ₁₆	A ₁₅	A ₁₀	A ₂	A ₁	A ₀	A ₃	IO ₀ & IO ₇	IO ₁ & IO ₆	IO ₂ & IO ₅	IO ₃ & IO ₄
0	0	0	0	0	0	0	BL ₃₈₄	BL ₂₅₆	BL ₁₂₈	BL ₀
0	0	0	0	0	0	1	BL ₃₈₅	BL ₂₅₇	BL ₁₂₉	BL ₁
0	0	0	0	0	0	1	BL ₃₈₆	BL ₂₅₈	BL ₁₃₀	BL ₂
0	0	0	0	0	0	1	BL ₃₈₇	BL ₂₅₉	BL ₁₃₁	BL ₃
0	0	0	0	0	1	0	BL ₃₈₈	BL ₂₆₀	BL ₁₃₂	BL ₄
0	0	0	0	0	1	0	BL ₃₈₉	BL ₂₆₁	BL ₁₃₃	BL ₅
0	0	0	0	0	1	1	BL ₃₉₀	BL ₂₆₂	BL ₁₃₄	BL ₆
0	0	0	0	0	1	1	BL ₃₉₁	BL ₂₆₃	BL ₁₃₅	BL ₇
•	•	•	•	•	•	•	•	•	•	•
1	1	1	1	1	0	0	BL ₅₀₈	BL ₃₈₀	BL ₂₅₂	BL ₁₂₄
1	1	1	1	1	0	1	BL ₅₀₉	BL ₃₈₁	BL ₂₅₃	BL ₁₂₅
1	1	1	1	1	1	0	BL ₅₁₀	BL ₃₈₂	BL ₂₅₄	BL ₁₂₆
1	1	1	1	1	1	1	BL ₅₁₁	BL ₃₈₃	BL ₂₅₅	BL ₁₂₇

Figure 21. 28F010 Bitline Decoding

X Address										Row
A ₁₄	A ₁₂	A ₇	A ₆	A ₅	A ₄	A ₁₃	A ₁₁	A ₉	A ₈	WL
0	0	0	0	0	0	0	0	0	0	XL ₀
0	0	0	0	0	0	0	0	0	1	XL ₁
0	0	0	0	0	0	0	0	1	0	XL ₂
0	0	0	0	0	0	0	0	1	1	XL ₃
0	0	0	0	0	0	0	1	0	0	XL ₄
0	0	0	0	0	0	0	1	0	1	XL ₅
0	0	0	0	0	0	0	1	1	0	XL ₆
0	0	0	0	0	0	0	1	1	1	XL ₇
0	0	0	0	0	0	1	0	0	0	XL ₈
0	0	0	0	0	0	1	0	0	1	XL ₉
0	0	0	0	0	0	1	0	1	0	XL ₁₀
0	0	0	0	0	0	1	0	1	1	XL ₁₁
0	0	0	0	0	0	1	1	0	0	XL ₁₂
0	0	0	0	0	0	1	1	0	1	XL ₁₃
0	0	0	0	0	0	1	1	1	0	XL ₁₄
0	0	0	0	0	0	1	1	1	1	XL ₁₅
0	0	0	0	0	1	1	1	1	1	XL ₁₆
0	0	0	0	0	1	1	1	1	0	XL ₁₇
0	0	0	0	0	1	1	1	0	1	XL ₁₈
0	0	0	0	0	1	1	1	0	0	XL ₁₉
0	0	0	0	0	1	1	0	1	1	XL ₂₀
0	0	0	0	0	1	1	0	1	0	XL ₂₁
0	0	0	0	0	1	1	0	0	1	XL ₂₂
0	0	0	0	0	1	1	0	0	0	XL ₂₃
0	0	0	0	0	1	0	1	1	1	XL ₂₄
0	0	0	0	0	1	0	1	1	0	XL ₂₅
0	0	0	0	0	1	0	1	0	1	XL ₂₆
0	0	0	0	0	1	0	1	0	0	XL ₂₇
0	0	0	0	0	1	0	0	1	1	XL ₂₈
0	0	0	0	0	1	0	0	1	0	XL ₂₉
0	0	0	0	0	1	0	0	0	1	XL ₃₀
0	0	0	0	0	1	0	0	0	0	XL ₃₁

Figure 22. 28F010 Wordline Decoding

X Address										Row
A ₁₄	A ₁₂	A ₇	A ₆	A ₅	A ₄	A ₁₃	A ₁₁	A ₉	A ₈	WL
0	0	0	0	1	0	0	0	0	0	XL32
•	•	•	•	•	•	•	•	•	•	•••
0	0	0	0	1	0	1	1	1	1	XL47
0	0	0	0	1	1	1	1	1	1	XL48
•	•	•	•	•	•	•	•	•	•	•••
0	0	0	0	1	1	0	0	0	0	XL63
0	0	0	1	0	0	0	0	0	0	XL64
•	•	•	•	•	•	•	•	•	•	•••
0	0	0	1	0	0	1	1	1	1	XL79
0	0	0	1	0	1	1	1	1	1	XL80
•	•	•	•	•	•	•	•	•	•	•••
0	0	0	1	0	1	0	0	0	0	XL95
1	1	1	1	1	0	0	0	0	0	XL992
•	•	•	•	•	•	•	•	•	•	•••
1	1	1	1	1	0	1	1	1	1	XL1007
1	1	1	1	1	1	1	1	1	1	XL1008
•	•	•	•	•	•	•	•	•	•	•••
1	1	1	1	1	1	0	0	0	0	XL1023

Figure 22. 28F010 Wordline Decoding (Continued)

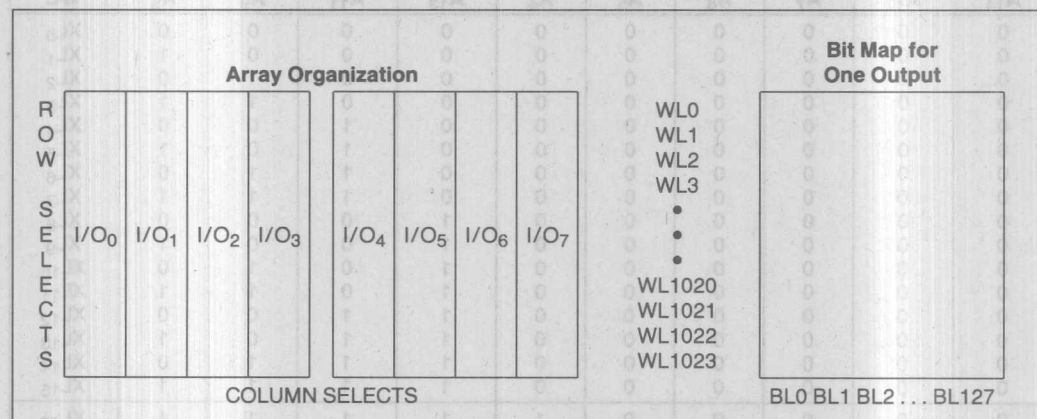


Figure 23. 28F010 Bit Map

Columns are number 0 through 511 beginning with the column nearest the X-decoder.
Outputs are grouped as follows:

Array Organization:							Left Half Array IO ₀ IO ₁ IO ₂ IO ₃ BL ₃₈₄ ← BL ₀		Right Half Array IO ₄ IO ₅ IO ₆ IO ₇ BL ₀ → BL ₃₈₄	
Address							Bitlines			
A ₁₆	A ₁₅	A ₁₀	A ₂	A ₁	A ₀	A ₃	IO ₀ & IO ₇	IO ₁ & IO ₆	IO ₂ & IO ₅	IO ₃ & IO ₄
0	0	0	0	0	0	0	BL ₃₈₄	BL ₂₅₆	BL ₁₂₈	BL ₀
0	0	0	0	0	0	1	BL ₃₈₅	BL ₂₅₇	BL ₁₂₉	BL ₁
0	0	0	0	0	1	0	BL ₃₈₆	BL ₂₅₈	BL ₁₃₀	BL ₂
0	0	0	0	0	1	1	BL ₃₈₇	BL ₂₅₉	BL ₁₃₁	BL ₃
0	0	0	0	1	0	0	BL ₃₈₈	BL ₂₆₀	BL ₁₃₂	BL ₄
0	0	0	0	1	0	1	BL ₃₈₉	BL ₂₆₁	BL ₁₃₃	BL ₅
0	0	0	0	1	1	0	BL ₃₉₀	BL ₂₆₂	BL ₁₃₄	BL ₆
0	0	0	0	1	1	1	BL ₃₉₁	BL ₂₆₃	BL ₁₃₅	BL ₇
•	•	•	•	•	•	•	•	•	•	•
1	1	1	1	1	0	0	BL ₅₀₈	BL ₃₈₀	BL ₂₅₂	BL ₁₂₄
1	1	1	1	1	0	1	BL ₅₀₉	BL ₃₈₁	BL ₂₅₃	BL ₁₂₅
1	1	1	1	1	1	0	BL ₅₁₀	BL ₃₈₂	BL ₂₅₄	BL ₁₂₆
1	1	1	1	1	1	1	BL ₅₁₁	BL ₃₈₃	BL ₂₅₅	BL ₁₂₇

Figure 24. 28F020 Bitline Decoding

X Address											Row
A ₁₇	A ₁₄	A ₁₂	A ₇	A ₆	A ₅	A ₄	A ₁₃	A ₁₁	A ₉	A ₈	WL
0	0	0	0	0	0	0	0	0	0	0	XL ₀
0	0	0	0	0	0	0	0	0	0	1	XL ₁
0	0	0	0	0	0	0	0	0	1	0	XL ₂
0	0	0	0	0	0	0	0	0	1	1	XL ₃
0	0	0	0	0	0	0	0	1	0	0	XL ₄
0	0	0	0	0	0	0	0	1	0	1	XL ₅
0	0	0	0	0	0	0	0	1	1	0	XL ₆
0	0	0	0	0	0	0	0	1	1	1	XL ₇
0	0	0	0	0	0	0	1	0	0	0	XL ₈
0	0	0	0	0	0	0	1	0	0	1	XL ₉
0	0	0	0	0	0	0	1	0	1	0	XL ₁₀
0	0	0	0	0	0	0	1	0	1	1	XL ₁₁
0	0	0	0	0	0	0	1	1	0	0	XL ₁₂
0	0	0	0	0	0	0	1	1	0	1	XL ₁₃
0	0	0	0	0	0	0	1	1	1	0	XL ₁₄
0	0	0	0	0	0	0	1	1	1	1	XL ₁₅
0	0	0	0	0	0	1	1	1	1	1	XL ₁₆
0	0	0	0	0	0	1	1	1	1	0	XL ₁₇
0	0	0	0	0	0	1	1	1	0	1	XL ₁₈
0	0	0	0	0	0	1	1	1	0	0	XL ₁₉
0	0	0	0	0	0	1	1	0	1	1	XL ₂₀
0	0	0	0	0	0	1	1	0	1	0	XL ₂₁
0	0	0	0	0	0	1	1	0	0	1	XL ₂₂
0	0	0	0	0	0	1	1	0	0	0	XL ₂₃
0	0	0	0	0	0	1	0	1	1	1	XL ₂₄
0	0	0	0	0	0	1	0	1	1	0	XL ₂₅
0	0	0	0	0	0	1	0	1	0	1	XL ₂₆
0	0	0	0	0	0	1	0	1	0	0	XL ₂₇
0	0	0	0	0	0	1	0	0	1	1	XL ₂₈
0	0	0	0	0	0	1	0	0	1	0	XL ₂₉
0	0	0	0	0	0	1	0	0	0	1	XL ₃₀
0	0	0	0	0	0	1	0	0	0	0	XL ₃₁

Figure 25. 28F020 Wordline Decoding

X Address											Row
A ₁₇	A ₁₄	A ₁₂	A ₇	A ₆	A ₅	A ₄	A ₁₃	A ₁₁	A ₉	A ₈	WL
0	0	0	0	0	1	0	0	0	0	0	XL32
•	•	•	•	•	•	•	•	•	•	•	•••
0	0	0	0	0	1	0	1	1	1	1	XL47
0	0	0	0	0	1	1	1	1	1	1	XL48
•	•	•	•	•	•	•	•	•	•	•	•••
0	0	0	0	0	1	1	0	0	0	0	XL63
0	0	0	0	1	0	0	0	0	0	0	XL64
•	•	•	•	•	•	•	•	•	•	•	•••
0	0	0	0	1	0	0	1	1	1	1	XL79
0	0	0	0	1	0	1	1	1	1	1	XL80
•	•	•	•	•	•	•	•	•	•	•	•••
0	0	0	0	1	0	1	0	0	0	0	XL95
0	1	1	1	1	1	0	0	0	0	0	XL992
•	•	•	•	•	•	•	•	•	•	•	•••
0	1	1	1	1	1	0	1	1	1	1	XL1007
0	1	1	1	1	1	1	1	1	1	1	XL1008
•	•	•	•	•	•	•	•	•	•	•	•••
0	1	1	1	1	1	1	0	0	0	0	XL1023
1	1	1	1	1	1	1	0	0	0	0	XL2016
•	•	•	•	•	•	•	•	•	•	•	•••
1	1	1	1	1	1	0	1	1	1	1	XL2031
1	1	1	1	1	1	1	1	1	1	1	XL2032
•	•	•	•	•	•	•	•	•	•	•	•••
1	1	1	1	1	1	1	0	0	0	0	XL2047

Figure 25. 28F020 Wordline Decoding (Continued)

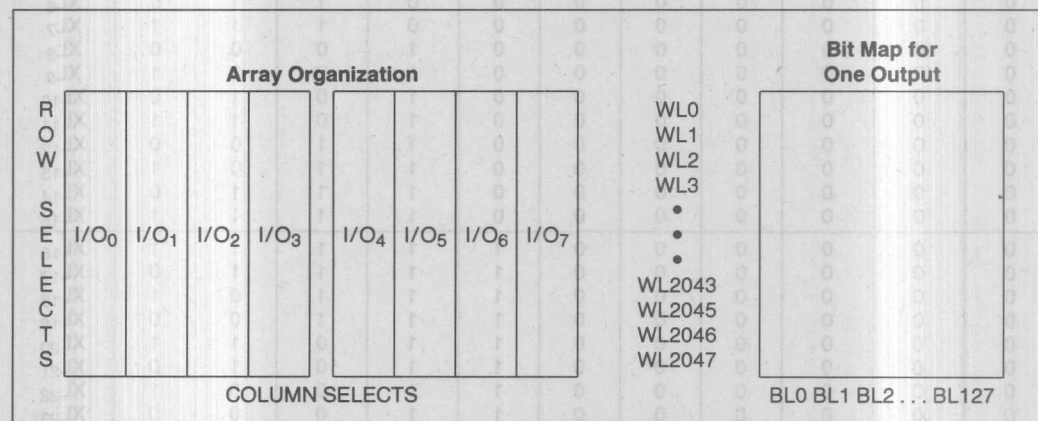


Figure 26. 28F020 Bit Map

Intel Flash Memory Evaluation Kit II (D, FLASHEVAL2) Product Brief

- Kit Contents**
- (1) PC AT*/PC XT* Add-In Driver Board with DIP ZIF Connector
 - (1) User's Manual
 - Sample Flash Memory Devices, Including:
 - (1) 28F256A
 - (1) 28F010
 - (1) 28F001BX-B
 - (1) 28F512
 - (1) 28F020
 - (1) 28F001BX-T
 - (1) 5.25" Floppy Disk with iFLASH2 Software
 - Technical Documentation Describing Intel's Flash Memory Products
 - Registration Card



Intel's Flash Memory Evaluation Kit II provides the system designer with a cost-effective prototyping tool for programming and erasing Intel Flash Memory products. Its software upgrade capability enables Intel to provide programming support for new flash memories coincident with their introduction. The Flash Memory Evaluation Kit's modular design provides compatibility with future Intel flash memory packages and form factors through hardware adapter upgrade modules. This kit is a significant enhancement to the Intel Flash Memory Evaluation Kit I, adding support for new devices and providing new software with easy user interface, additional capabilities and on-line help.

Kit Description

The Intel Flash Memory Evaluation Kit II is a PC-driven flash memory programming solution. With this kit, a system designer can program and erase Intel's flash memory components directly, and flash memory cards, SIMMs and advance packaged components using separately available hardware adapters.

The kit's Users Guide provides software and hardware installation instructions and an overview of software features.

*PC-XT and PC-AT are trademarks of International Business Machines, Inc.

Technical documentation includes device datasheets, application notes and reliability information. Together, these documents provide a complete description of the technology and important design considerations.

Technical documentation includes device datasheets, application notes and reliability information. Together, these documents provide a complete description of the technology and important design considerations.

SMALL OUTLINE PACKAGE PHYSICAL DIMENSIONS

1.0 Case Outlines for Intel's 32-, 40-, 56-Lead TSOP and 44-Lead PSOP Packages

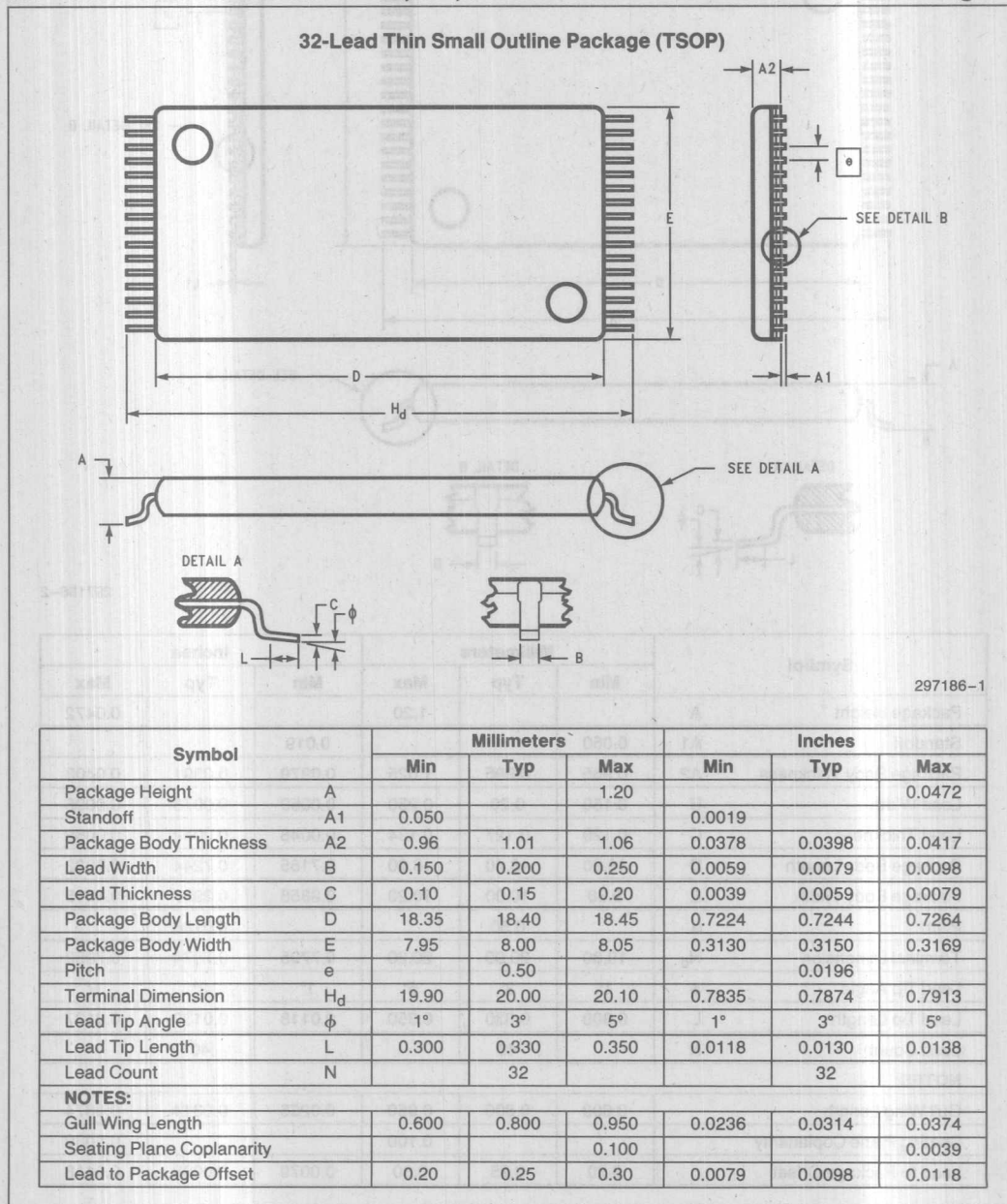


Figure 1-1. 32-Lead TSOP Package Drawing and Specifications

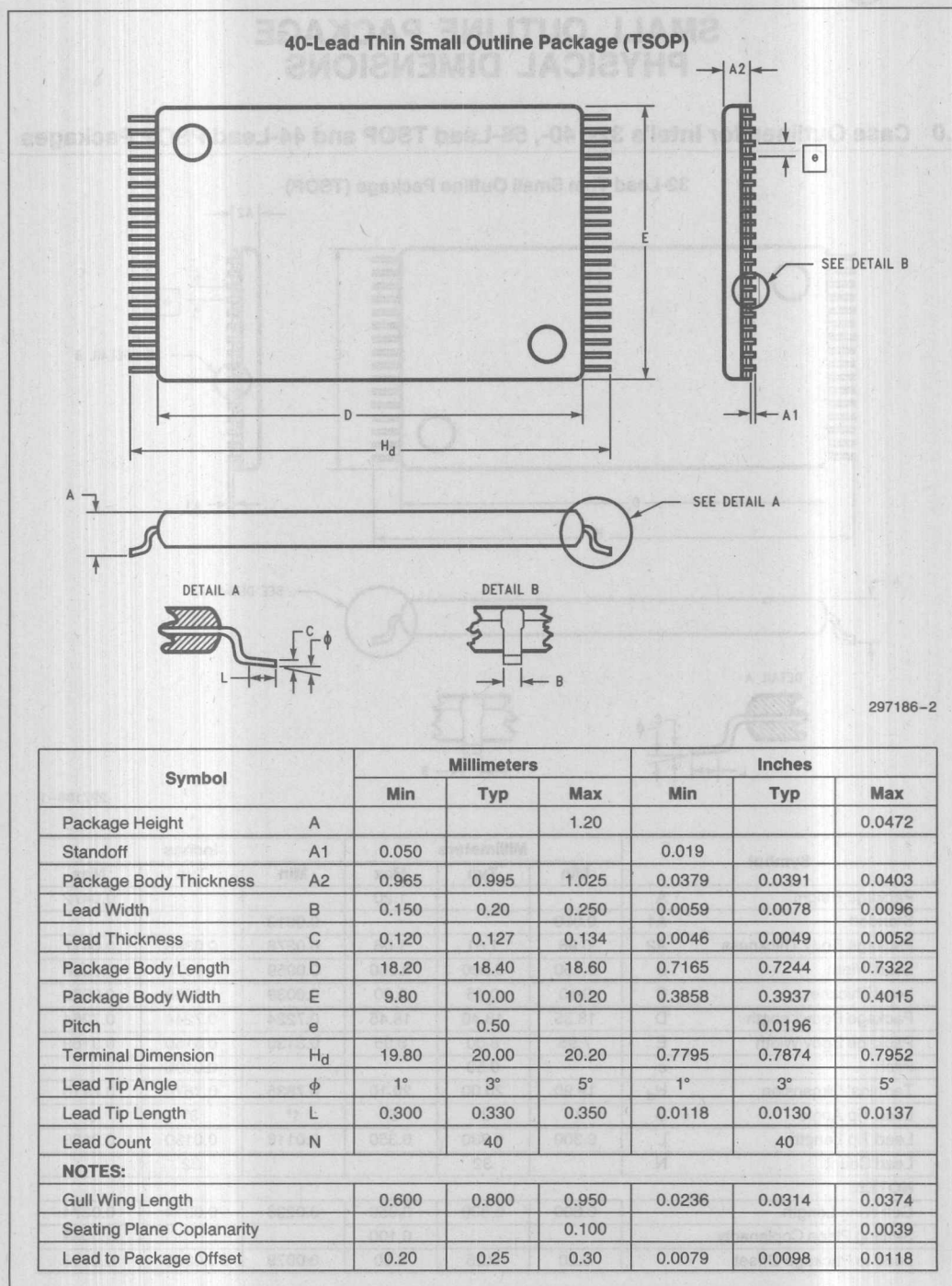
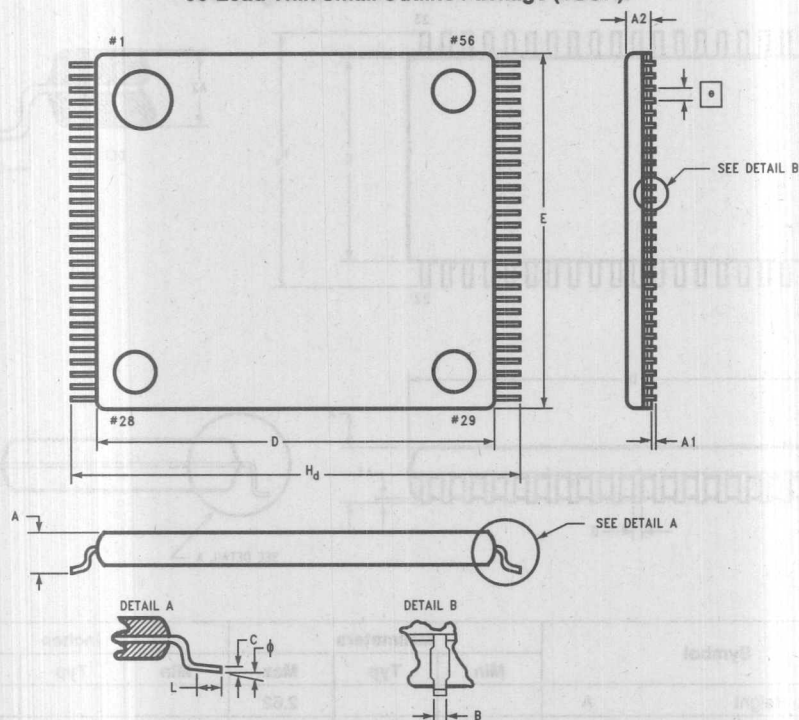


Figure 1-2. 40-Lead TSOP Physical Dimensions and Specifications

56-Lead Thin Small Outline Package (TSOP)



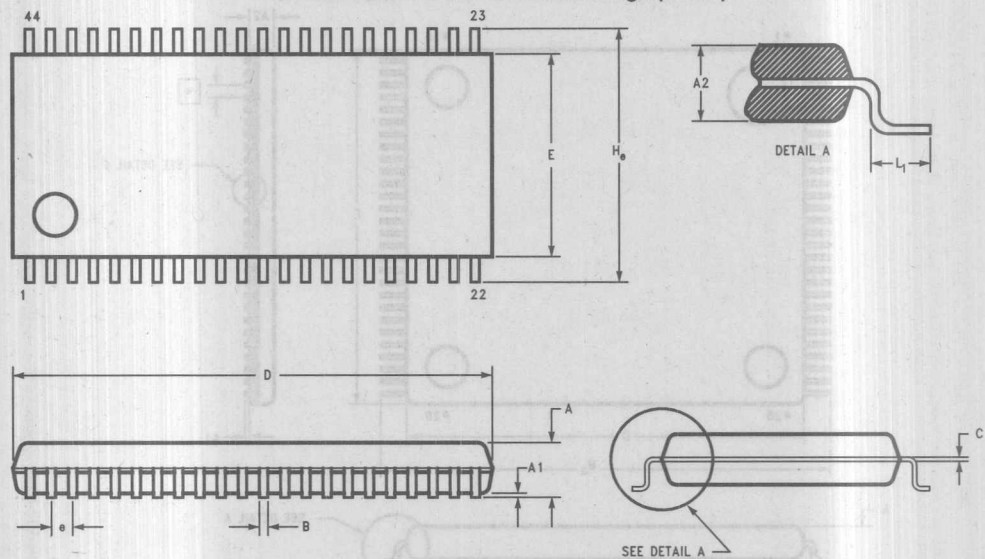
297186-3

5

Symbol		Millimeters			Inches		
		Min	Typ	Max	Min	Typ	Max
Package Height	A			1.20			0.0472
Standoff	A1	0.050			0.0019		
Package Body Thickness	A2	0.965	0.995	1.025	0.0379	0.0391	0.0403
Lead Width	B	0.100	0.150	0.200	0.0039	0.0059	0.0079
Lead Thickness	C	0.120	0.127	0.134	0.0047	0.0049	0.0053
Package Body Length	D	18.20	18.40	18.60	0.7165	0.7244	0.7322
Package Body Width	E	13.80	14.00	14.20	0.5433	0.5511	0.5590
Pitch	e		0.50			0.0196	
Terminal Dimension	Hd	19.80	20.00	20.20	0.7795	0.7874	0.7952
Lead Tip Angle	φ	1°	3°	5°	1°	3°	5°
Lead Tip Length	L	0.300	0.330	0.350	0.0118	0.0130	0.0137
Lead Count	N		56			56	
NOTES:							
Gull Wing Length		0.600	0.800	0.950	0.0236	0.0314	0.0374
Seating Plane Coplanarity				0.100			0.0039
Lead to Package Offset		0.20	0.25	0.30	0.0079	0.0098	0.0118

Figure 1-3. 56-Lead TSOP Package Drawings and Specifications

44-Lead Plastic Small Outline Package (PSOP)



297186-4

Symbol		Millimeters			Inches		
		Min	Typ	Max	Min	Typ	Max
Package Height	A			2.62			0.103
Standoff	A1	0.13	0.225	0.35	0.005	0.009	0.013
Package Body Thickness	A2	2.17	2.30	2.45	0.085	0.091	0.097
Lead Width	B	0.35	0.40	0.50	0.014	0.016	0.020
Lead Thickness	C	0.13	0.150	0.20	0.005	0.006	0.008
Package Body Length	D		28.20	28.70		1.100	1.130
Package Body Width	E	13.10	13.30	13.50	0.516	0.524	0.531
Pitch	e		1.27			0.050	
Terminal Dimension	H _e	15.70	16.00	16.30	0.618	0.630	0.642
Lead Tip Angle	φ			8°			8°
Lead Tip Length	L1	0.75	0.80	0.85	0.029	0.032	0.033
Lead Count	N		44			44	
NOTES:							
Gull Wing Length		1.30	1.35	1.40	0.051	0.053	0.055
Seating Plane Coplanarity				0.10			0.004

Figure 1-4. 44-Lead PSOP Physical Dimensions and Specifications

2.0 SOP Board Footprints

A typical land pad diagram for the 32-Lead TSOP package is shown in Figure 2-1.

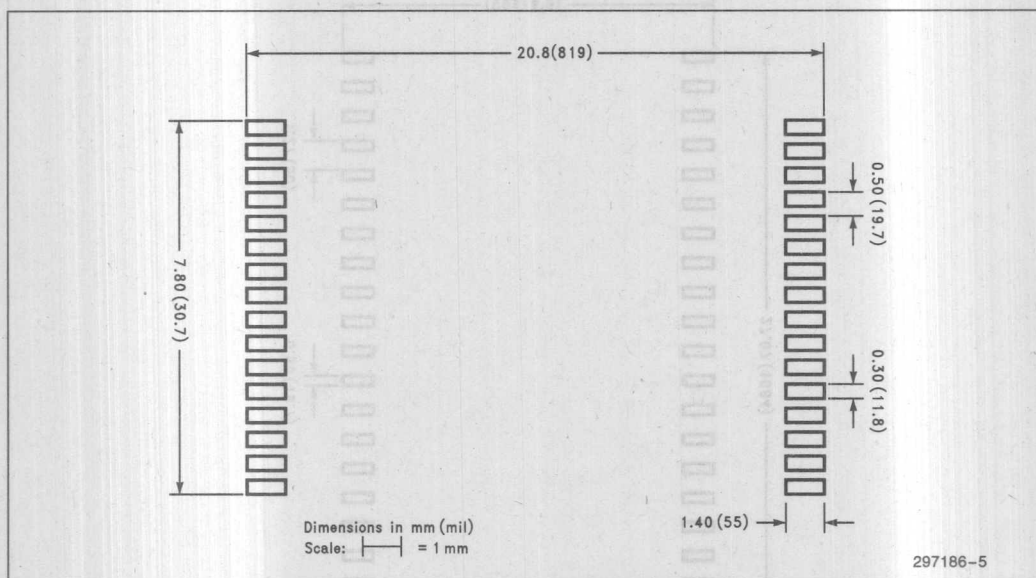


Figure 2-1. Typical TSOP Land Pad Diagram

Similar land pad diagrams can be constructed for the 40-lead and 56-lead TSOP packages. The 40-lead land pad diagram can be constructed by adding four leads to both sides of the 32-lead land pad diagram, while maintaining the spacing between the lead footprints. The total footprint width becomes 9.80mm instead of 7.80mm used with the 32-lead package. With the 56-lead package, 12 leads are added to both sides of the 32-lead land pad diagram, while maintaining the spacing between the lead footprints. The total footprint width becomes 13.80mm instead of the 7.80mm used with the 32-lead package.

A typical land pad diagram for the 44-Lead PSOP is shown in Figure 2-2.

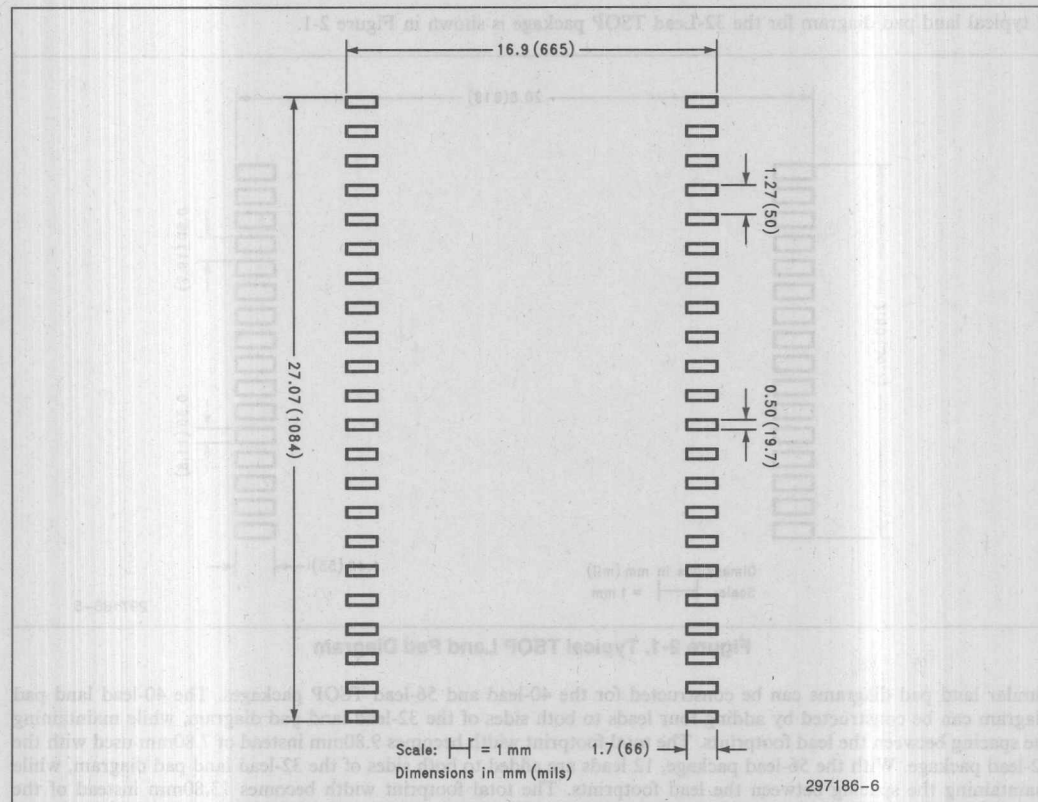


Figure 2-2. Typical PSOP Land Pad Diagram

3.0 SOP Component Volume and Weight

Table 3-1 shows the component volume and weight of the SOP package family.

Table 3-1. SOP Component Weight and Volume

Package	Max Height	Max Volume	Average Weight
32-Lead TSOP	1.20 mm	194.2 mm ³	0.37 gms
40-Lead TSOP	1.20 mm	247.2 mm ³	0.47 gms
56-Lead TSOP	1.20 mm	344.2 mm ³	0.65 gms
44-Lead TSOP	2.62 mm	1,225.7 mm ³	186 gms